MRC mission research corporation

WILDLAND FIRE MODELING FOR PLANNING SUPPORT

VOLUME III -- A PROTOTYPE SEGMENTED-PERIMETER WILDLAND FIRE PROPAGATION MODEL

MRC Report No. 7611-3-876

Submitted to:

USDA Forest Service
Contract No. 21-393

Submitted by:

Mission Research Corporation
P.O. Drawer 719
Santa Barbara, CA 93102

Prepared by:

J. C. Sanderlin
J. M. Sunderson

August 1976

TABLE OF CONTENTS CONTINUED

VOLUME III

1.0	A PROTOTYPE SEGMENTED-PERIMETER WILDLIFE FIRE PROPAGATION MODEL	1
2.0	FUEL CHARACTERISTICS	1
2.1	COMPARISON OF MEASURED PERIMETERS AND SIMULATED PERIMETERS BASED ON TABULATED FUEL CHARACTERISTICS	5
2.2	COMPARISON OF MEASURED PERIMETERS WITH SIMULATED PERIMETERS BASED ON PROTOTYPE FUEL CHARACTERISTICS ALGORITHMS	12
3.0	LOCAL SPREAD RATE PATTERN	19
4.0	LOCATION OF FIRE ORIGIN AND REDISTRIBUTION OF PERIMETER POINTS	21
4.1	LOCATION OF FIRE ORIGIN	21
4.2	PERIMETER POINT REDISTRIBUTION	24
5.0	CONCLUSIONS	29
6.0	SEGMENTED PERIMETER PROPAGATION MODEL	32
6.1	SYSTEM MANAGER (RADSM)	32
6.2	PERIMETER POINT REDISTRIBUTION (REDIST)	40
6.3	PERIMETER POINT ADVANCE (PPTADV)	45
6.4	PERIMETER GROWTH (RADPER)	49
6.5	PERIMETER LOOP REMOVAL (MASSAG)	56
6.6	TIME INCREMENT CALCULATION (DLTRM)	66
6.7	INTERSECTION CALCULATION (INTERS)	68
6.8	SPIRAL INDEX CALCULATION (SPIND AND ISPIND)	72
6.9	SCATTER STORAGE FUNCTION (ISS)	76
6.10	RESOLUTION ADJUSTMENTS (RESOLU)	78
6.11	SELECT RIGHT LOOP (RTLOOP)	81

TABLE OF CONTENTS CONTINUED

6.12 POSITIVE POLAR ANGLE (THETAO) 83
6.13 ERROR OUTPUT (DOUT) 85
6.14 RESOLUTION SETUP (RESSET) 87

LIST OF FIGURES

Figure 1	Calculated Potrero Fire Perimeters.	2
Figure 2	Measured Potrero Fire Perimeters.	3
Figure 3	Potrero Fire Simulation With Coastal Sage Represented by Chamise.	6
Figure 4	Potrero Fire Simulation With Coastal Sage Represented by Brush..	7
Figure 5	Overlay of Figure 1 (Dark) and Figure 4 (Light).	8
Figure 6	Spread Rates From Measured Potrero Fire Perimeters	10
Figure 7	Rates of Spread From Simulated Potrero Fire Based on Tabulated Fuel Characteristics	11
Figure 8	Length and Width of Measured Perimeters and Simulated Perimeters based upon Tabulated Fuel Characteristics.	13
Figure 9	Rates of Spread From Simulated Potrero Fire Based on Prototype Fuel Characteristics Algorithms.	16
Figure 10	Length and Width of Measured Perimeters and Simulated Perimeters Based on Fuel Characterisitics Algorithms.	18
Figure 11	Elliptical, Local Spread Rate Patterns for Several Values of Wind Speed.	20
Figure 12	Local Component Spread Pattern.	22
Figure 13	Potrero Fire Simulation for Local Component Spread Rate Pattern.	23
Figure 14	Potrero Fire Simulation with Origin Displaced 120 Ft. to Right.	25
Figure 15	Potrero Fire Simulation With Dynamic Perimeter Point Distribution.	28

LIST OF FIGURES CONTINUED

Figure 16	Spread Rates From Simulated Potrero Fire with Perimeter Point Redistribution.	30
Figure 17	Length and Width of Measured Perimeters and Simulated Perimeters Using Perimeter Point Redsitribution.	31
Figure 18	Hierarchy of Radial Spread Model Routines.	36
Figure 19	Radial System Manager.	37
Figure 20	Perimeter Point Redistribution.	42
Figure 21	Perimeter Point Advances.	46
Figure 22	Radial Perimeter Growth.	50
Figure 23	Loop Removal.	58
Figure 24	Intersection Calculation.	70
Figure 25	Example of a Grid with Spiral Indices.	73

A PROTOTYPE SEGMENTED-PERIMETER WILDLAND FIRE PROPAGATION MODEL

1.0

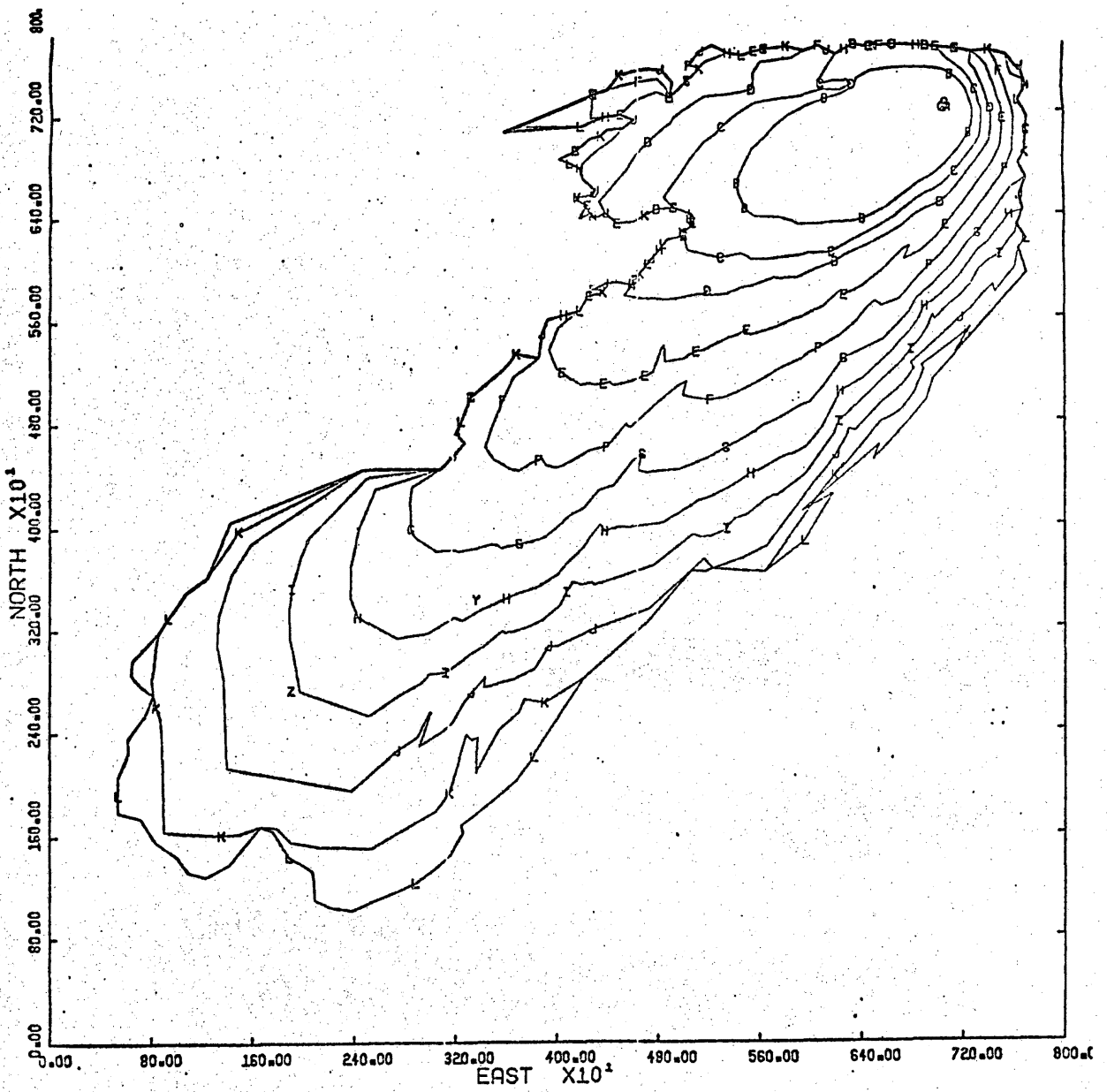
Progress in the development of a prototype segmented-perimeter wildland fire propagation model during the current contract period consists primarily of:

- o inclusion of algorithms for estimating fuel characteristics required for the calculation of spread rates.
- o a brief examination of an alternative, local spread rate pattern.
- o inclusion of dynamic redistribution of perimeter points.
- o prototype model documentation.

2.0 FUEL CHARACTERISTICS

During the current contract period, prototype algorithms for estimating the fuel characteristics required by the rate of spread model[1] have been evaluated and documented.[2] The fuel characteristics algorithms estimate the fuel properties required by the spread rate model in terms of specified fuel type and age.[3] The prototype algorithms currently admit five fuel type classifications commensurate with Southern California fuels (short and long grass, brush, chamise and mixed chaparral).

During the previous contract period a rudimentary form of a segmented propagation model was exercised against a historical Southern California fire.[3] In this case the required fuel characteristics were obtained from a fuel characteristics table prepared by the USDA Forest Service Fire Research Laboratory at Riverside, California, from a post-fire survey of the burned area. Time sequences of perimeters estimated by the rudimentary segmented propagation model[3] and perimeters of the actual fire[4] (from infrared photographs) are included here for reference purposes as Figures 1 and 2, respectively.



LEGEND

A = 12.58 HR
B = 13.34 HR
C = 13.48 HR
D = 14.02 HR
E = 14.16 HR
F = 14.30 HR
G = 14.44 HR
H = 14.57 HR
I = 15.10 HR
J = 15.25 HR
K = 15.38 HR
L = 15.52 HR

100% Wind

Approx. 40 mph

Figure 1

Calculated Potrero Fire Perimeters

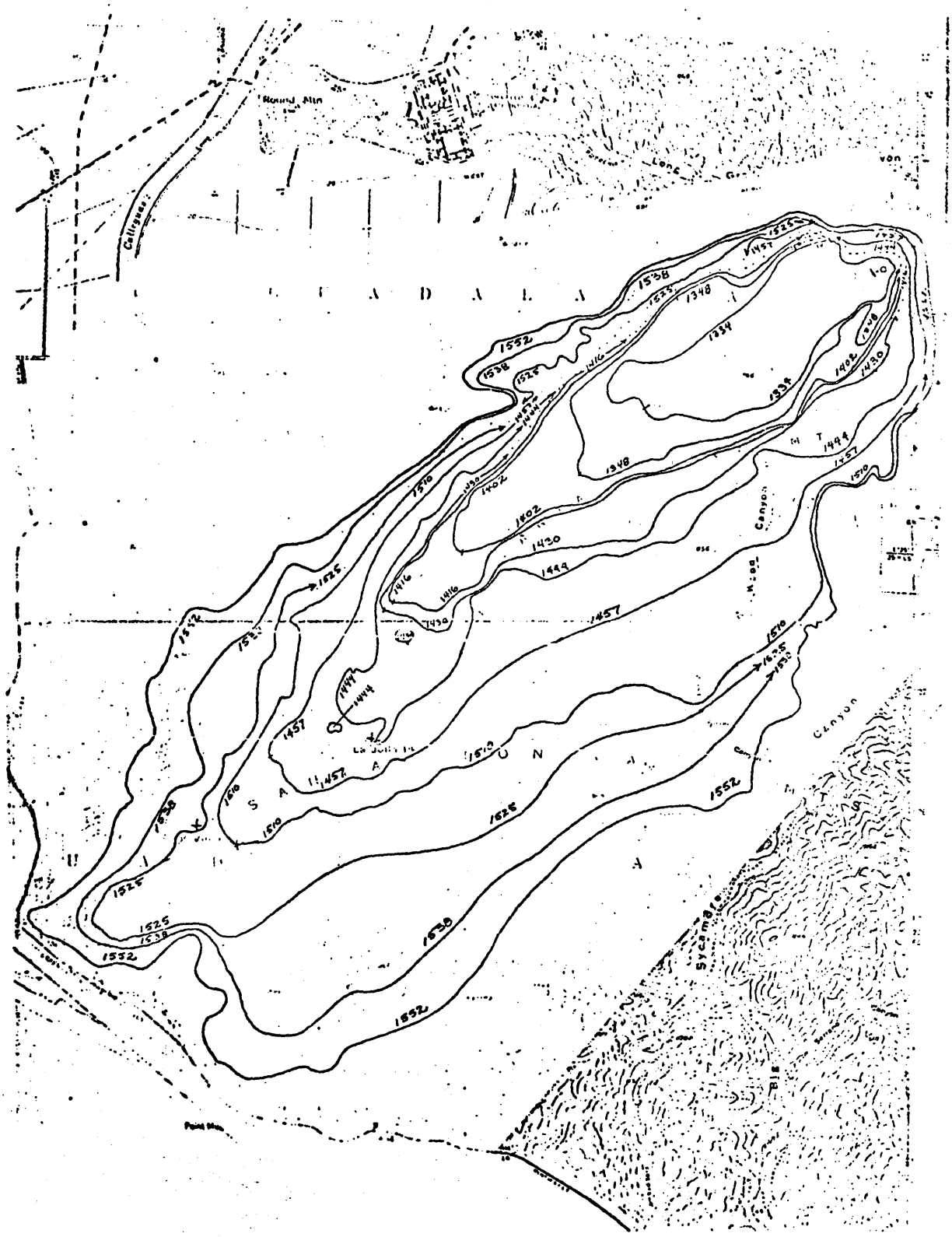


Figure 2

Measured Potrero Fire Perimeters

Ideally, the replacement of tabulated fuel characteristics derived from observation by fuel characteristics derived from mathematical models based on fuel type and age should produce no change in the estimated fire history as described by a time sequence of calculated fire perimeters. Such an idealization is not possible in this case for two reasons:

1. The principal fuel type (i.e., coastal sage) extant in the area of the historical fire of interest is indigeneous to the Southern California coastal areas, and these areas are not of prime importance in consideration of Southern California wildland fires. Thus, coastal sage is not one of the 5 fuel types whose characteristics are currently described by the prototype fuel characteristics algorithms.

2. The prototype fuel characteristics algorithms do not include site dependent variables such as elevation, slope, aspect, solar insolation, rainfall, history, etc., which may affect fuel characteristics. Thus, the results of the prototype fuel characteristics model will not necessarily reproduce the contents of a table of fuel characteristics derived from observation.

The description of a fire history by a sequence of calculated perimeters is considered to be an estimate. That is, the gross features (general perimeter shape) and trends (direction and rate of growth at the head, flanks, etc.) exhibited by the sequence of calculated perimeters should approximate measured conditions; however, the detailed shape or precise location of any calculated perimeter or perimeter segment should not be considered significant. For these reasons it was considered sufficient that the sequence of calculated perimeters employing fuel characteristics derived from mathematical models should agree "reasonably well" with the sequence of perimeters employing tabulated fuel characteristics. For the present purposes "reasonable correspondence" is defined qualitatively and quantitatively by a comparison of rates of growth at the head and flanks of the fire, together with the overall perimeter shapes.

Figure 3 presents a simulation of the Potrero fire in which Chamise was used to represent coastal sage. A superficial comparison of Figure 3 with Figure 1 reveals that the simulated fire of Figure 3 spreads to the southwest much too rapidly. Thus, chamise may be considered a poor representation of coastal sage.

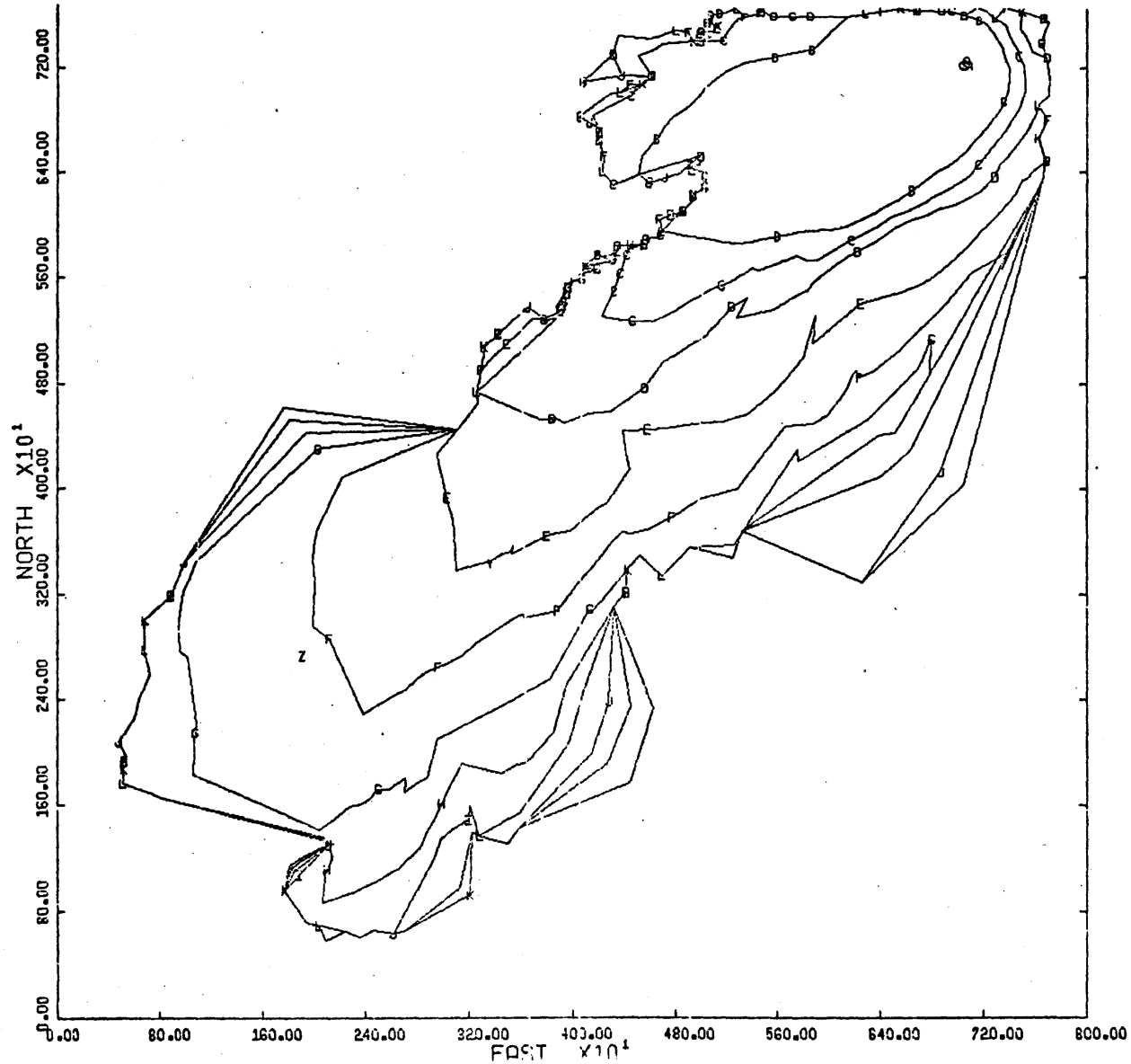
Figure 4 presents a simulation of the Potrero fire in which coastal sage is represented by brush. A qualitative comparison of Figures 1 and 4, such as the overlay of Figure 5, shows that the general shape of the perimeters, the rate of growth at the head of the fire and the rates of growth at the flanks are all reasonably consistent. A somewhat more quantitative comparison is described below.

2.1 COMPARISON OF MEASURED PERIMETERS AND SIMULATED PERIMETERS BASED ON TABULATED FUEL CHARACTERISTICS

Figure 6 presents spread rates calculated from data taken from the measured perimeters of Figure 2.[3,4] Spread rates in these different directions are shown in Figure 6. Southwest is the general direction of the head of the fire; southeast is the general direction of the only flank on which the fire could spread, and south is the direction of major spread for times greater than about 15:25. It can be seen in Figure 6 that spread rate to the southwest is shown from 13:48 to 15:42, spread rate to the southeast is shown from 14:20 to 15:00 and that spread rate to the south is shown from 14:34 to 15:42.

Figure 7 shows spread rates calculated from data taken from the simulated perimeters of Figure 1, based upon tabulated fuel characteristics. Comparison of Figure 6 and Figure 7 reveals:

- The southwest spread rates are similar in form, except the large measured peak between 15:10 and 15:25 is absent from Figure 7. The absence of this peak is somewhat compensated by the higher

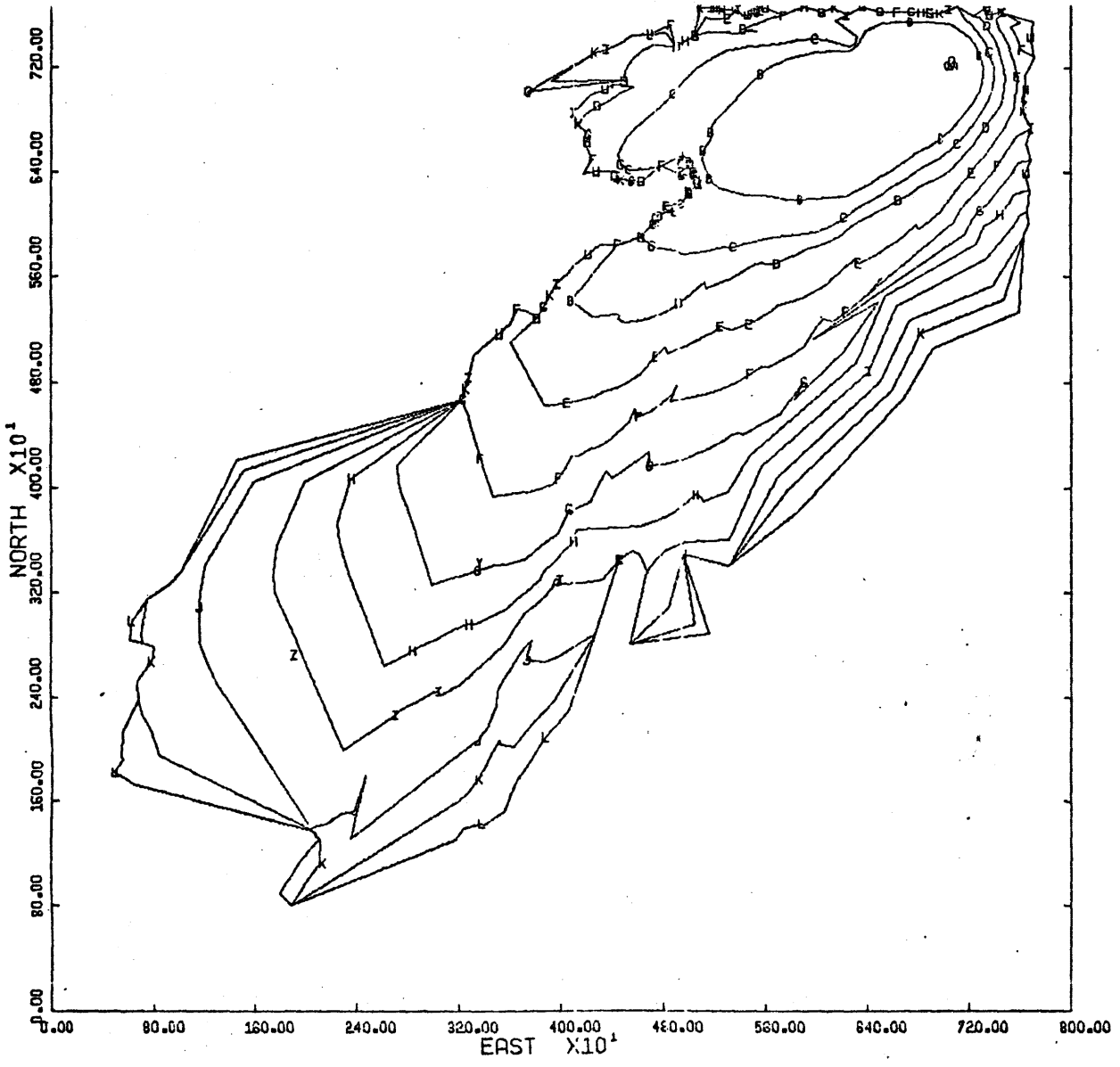


LEGEND

A = 12.58 HR
B = 13.34 HR
C = 13.48 HR
D = 14.02 HR
E = 14.16 HR
F = 14.30 HR
G = 14.44 HR
H = 14.57 HR
I = 15.10 HR
J = 15.25 HR
K = 15.38 HR
L = 15.52 HR

Figure 3

Potrero Fire Simulation With Coastal Sage Represented by Chamise

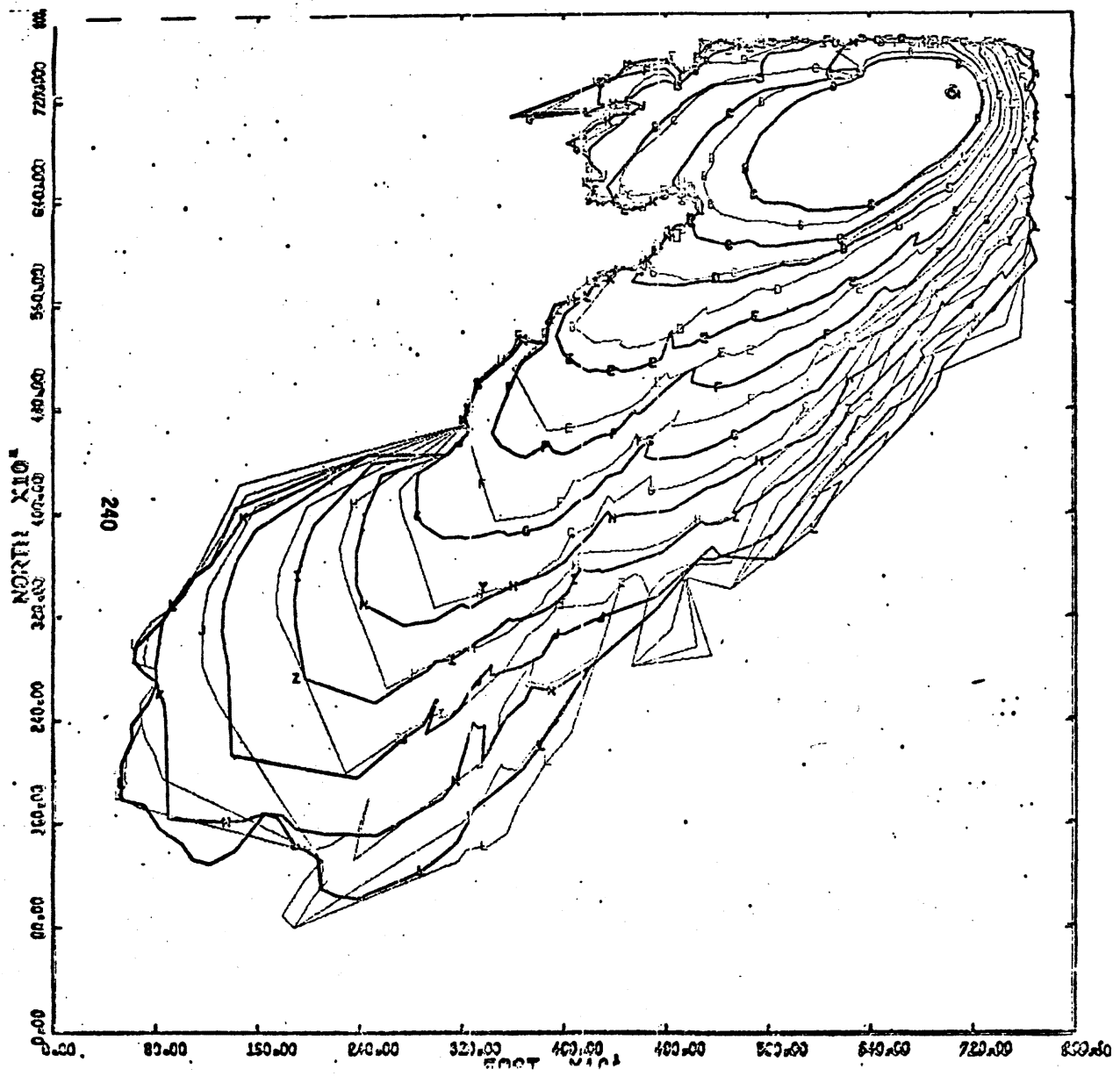


LEGEND

A = 12.58 HR
B = 13.34 HR
C = 13.48 HR
D = 14.02 HR
E = 14.16 HR
F = 14.30 HR
G = 14.44 HR
H = 14.57 HR
I = 15.10 HR
J = 15.25 HR
K = 15.38 HR
L = 15.52 HR

Figure 4

Potrero Fire Simulation with Coastal Sage Represented by Brush



LEGEND

A = 12.58 HR
B = 13.34 HR
C = 13.48 HR
D = 14.02 HR
E = 14.16 HR
F = 14.30 HR
G = 14.44 HR
H = 14.57 HR
I = 15.10 HR
J = 15.25 HR
K = 15.38 HR
L = 15.52 HR

100% Wind

Approx. 40 mph

Figure 5

Overlay of Figure 1 (Dark) and Figure 4 (Light)

average values of Figure 7 between 14:00 and 15:00, which result in the average spread rate over the entire interval from 13:34 to 15:44 in Figure 7 being 93 percent of the average spread rate in Figure 6. Thus, the length of the fire toward the southwest is generally consistent between Figures 1 and 2, although a significant discrepancy may be found at any particular time.

- The southeast spread rate of Figure 7 exhibits both incorrect values and incorrect slope. However, the initial value is high while the final value is low, so that over the interval from 14:20 to 15:00 the average simulated value of Figure 7 is about 71 percent of the average measured value. Thus, the perimeters of Figure 1 are too wide early in the fire (i.e., upper right-hand corner), and not wide enough late in the fire (i.e., lower left-hand corner). It is worth noting that the southeastward expansion of the simulated perimeters of Figure 1 is halted by a region defined in the observational derived fuel table as barren land. This barren area is located in the lower center of perimeters J, K and L. It should also be noted that this is a region of maximum southeast spread rate for the measured perimeters of Figure 2. It is considered possible that what may have appeared under post-burn observation as barren ground could have actually been sparsely covered by grass, which under the meteorological conditions at the time of the fire, would have resulted in a rapid southeastward spread in this area.

- Spread rate to the south shown in Figure 7 is initially low and remains low essentially throughout the interval. Additionally the large peak shown in Figure 6 at 15:30 is completely absent from Figure 7. This peak is a result of the southward run between 15:25 and 15:28, which is clearly shown in the lower left-hand corner of Figure 2, but which is only weakly evidenced in Figure 1.

Figure 6. Spread Rates from Measured Potrero Fire Perimenters

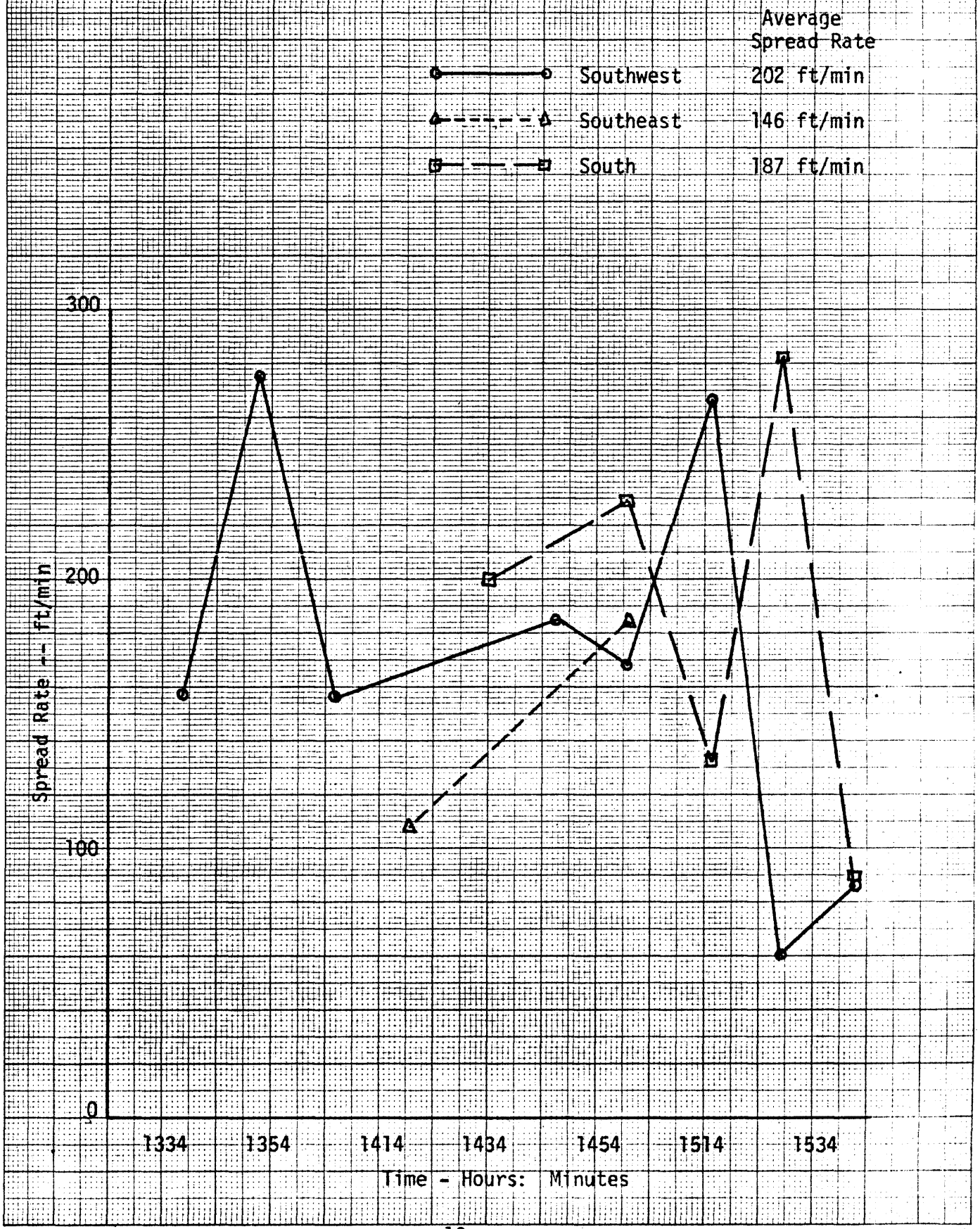
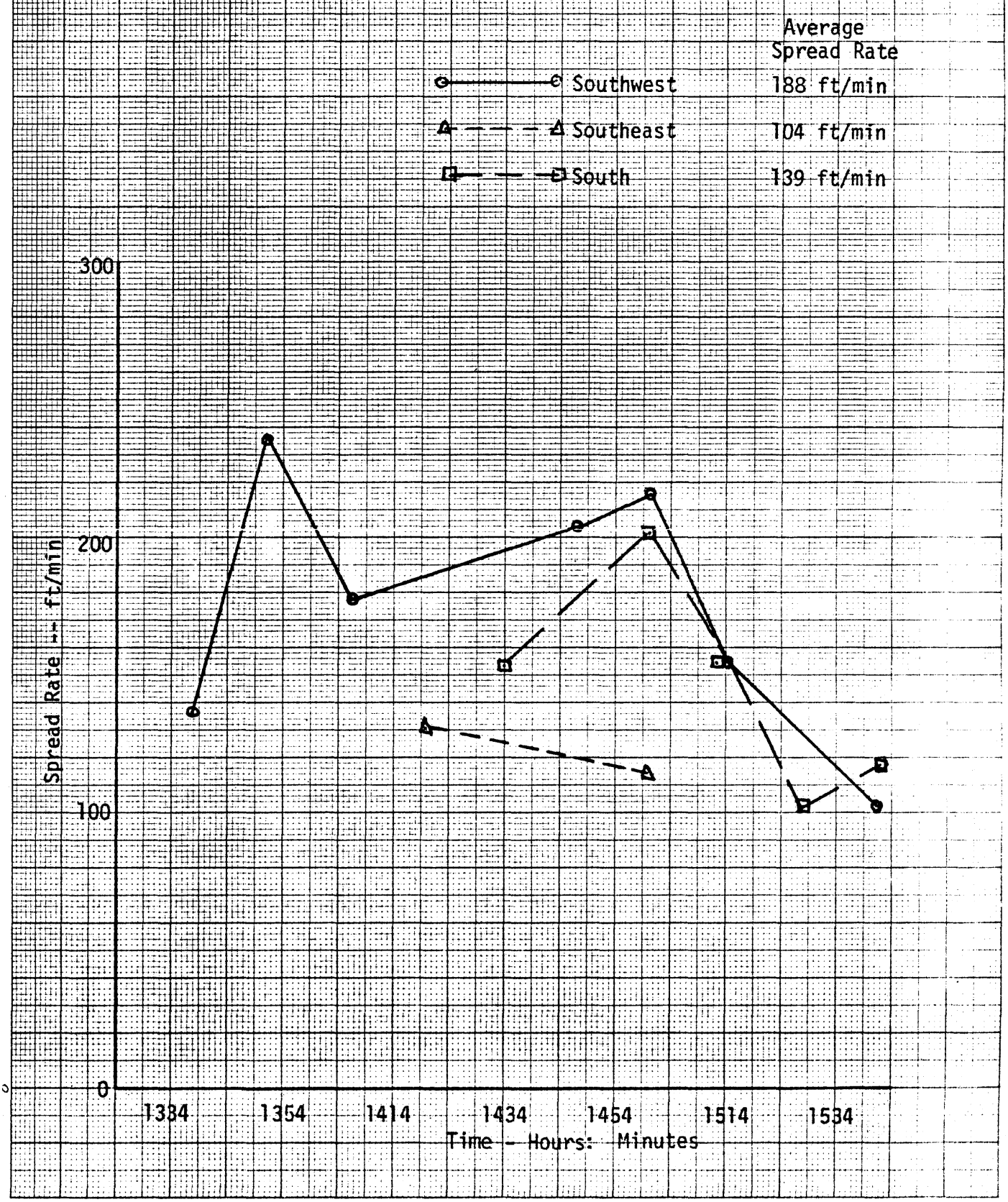


Figure 7. Rates of Spread from Simulated Potrero Fire Based on Tabulated Fuel Characteristics



As a result, the average simulated spread rate to the south is only about 75 percent of the average southward spread rate for the measured perimeters.

Figure 8 presents the lengths and widths of the measured perimeters and the simulated perimeters based on tabulated fuel characteristics. The length of the fire plotted in Figure 8 is the distance from the head of the 13:34 perimeter to the heads of successive perimeters. The width of the fire plotted in Figure 8 is the width of each perimeter of both sets measured at the widest point of the measured perimeters.* It can be seen in the upper part of Figure 8 that although the early deviations in measured rate of change of length with respect to time are not evidenced by the simulated results, that for times greater than about 14:30 the simulated and measured fire lengths agree well. The comparison of fire widths depicted in the lower part of Figure 8 shows that prior to about 15:00 the simulated perimeters are wider than the measured perimeters, as shown in the upper right-hand corners of Figures 1 and 2, respectively. For times greater than about 15:00 the widths of the simulated perimeters, at the point of measurement, are narrower than the measured perimeters as shown in the lower left-hand corners of Figures 1 and 2, respectively.**

2.2 COMPARISON OF MEASURED PERIMETERS WITH SIMULATED PERIMETERS BASED ON PROTOTYPE FUEL CHARACTERISTICS ALGORITHMS

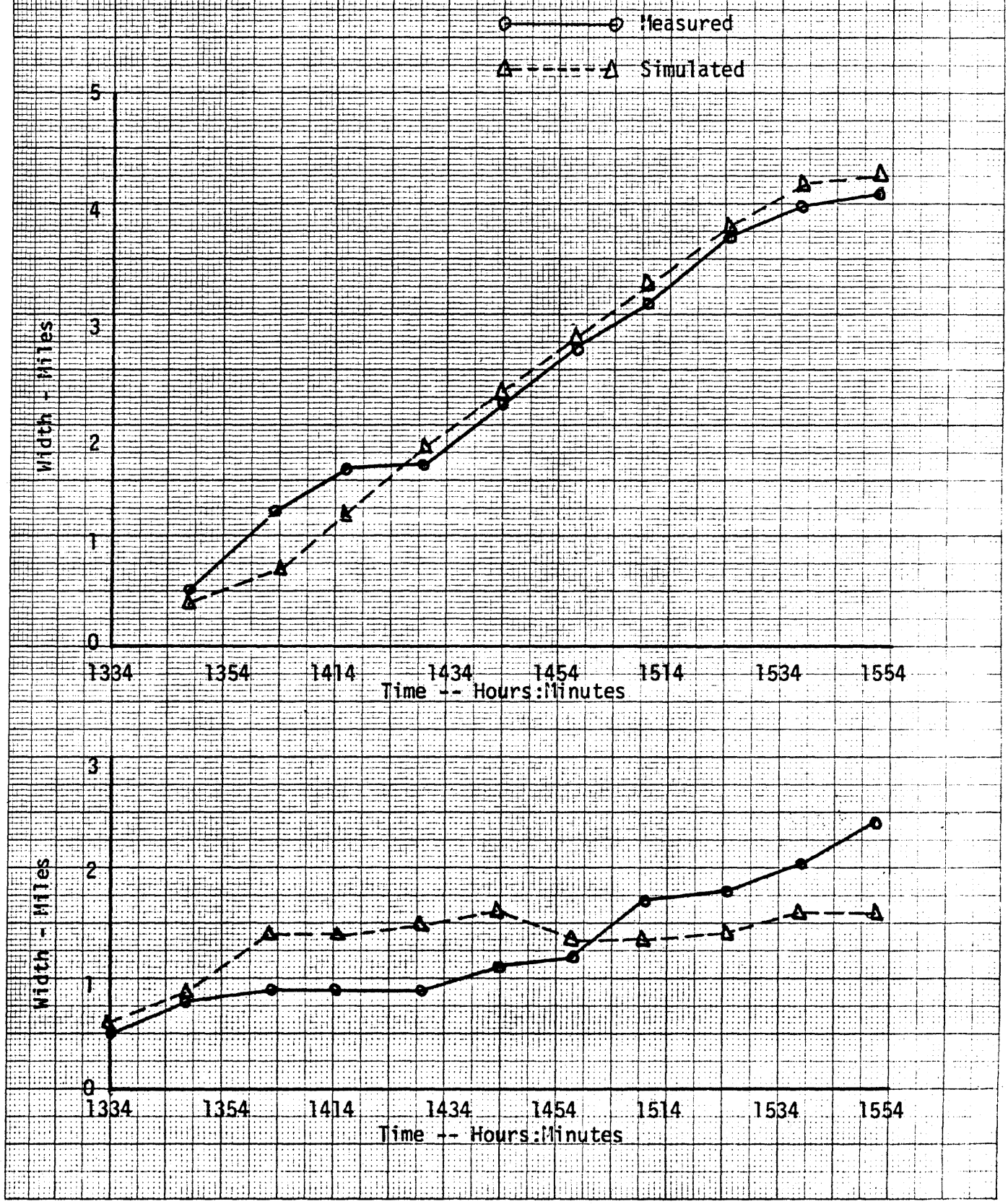
Figure 9 presents rates of spread for the simulated Potrero fire where tabulated fuel characteristics have been replaced by fuel characteristics algorithms. The fire spread rate model requires entry of a set of fuel characteristics such as fuel depth, fuel particle area-to-volume ratio, etc.[1]

---

*These comments also apply to the curves of Figure 10 and Figure 17. Since the point of measurement follows the location of greatest width on the measured perimeters, the plotted width of the simulated perimeters may decrease!

**Maximum width of the measured perimeters shifts from the upper right-hand corner of Figure 2 to the lower left-hand corner for times greater than about 15:25. This does not occur for the simulated perimeters.

Figure 8. Length and Width of Measured Perimeters and Simulated Perimeters based upon Tabulated Fuel Characteristics



These fuel characteristics are generally different for different fuel types and for each fuel type may vary with fuel age. The prototype fuel characteristics algorithms provide an estimate of each required fuel characteristic as a function of fuel type and age.* Thus, the simulated perimeters based on fuel characteristics algorithms employ the same fuel type-age map that was employed for the simulation based on tabulated fuel characteristics; however, the detailed fuel characteristics required by the spread rate model are generated by mathematical models instead of being obtained from a look-up table.

The current, prototype fuel characteristics algorithms provide for five fuel types common in Southern California wildland fires; short grass, long grass, brush, chamise, and mixed chaparral. Coastal sage, the principal fuel type in the Potrero fire, is not one of these fuels. Brush was chosen as the fuel type within the capability of the prototype fuel characteristics models that was most representative of coastal sage.

Comparison of Figure 6 (Spread rates for measured perimeters) and Figure 9 (spread rates for simulated perimeters based on prototype fuel character algorithms) reveals:

- Spread rate to the southwest is initially the same, although the measured peaks at 13:50 and 15:16 are absent from the simulated results. The simulated spread rate is about 10 percent higher between 14:14 and 15:10, thus offsetting, to some degree, the absence of the two peaks.** The average simulated spread rate over the interval is about 80 percent of the average spread rate for the measured perimters.

---

*The characteristics of a fuel type at given age may also vary due to site dependent characteristics such as moisture or soil type; however, the prototype algorithms do not consider these effects.

**Examination of the measured length shown in the upper part of Figure 8 at about 13:50 and 15:10 shows that the integrated effect of these two large measured peaks in southwest spread rate is small.

Comparison of Figure 6 (Spread rates for measured perimeters) and Figure 9 (spread rates for simulated perimeters based on prototype fuel characteristics algorithms) reveals:

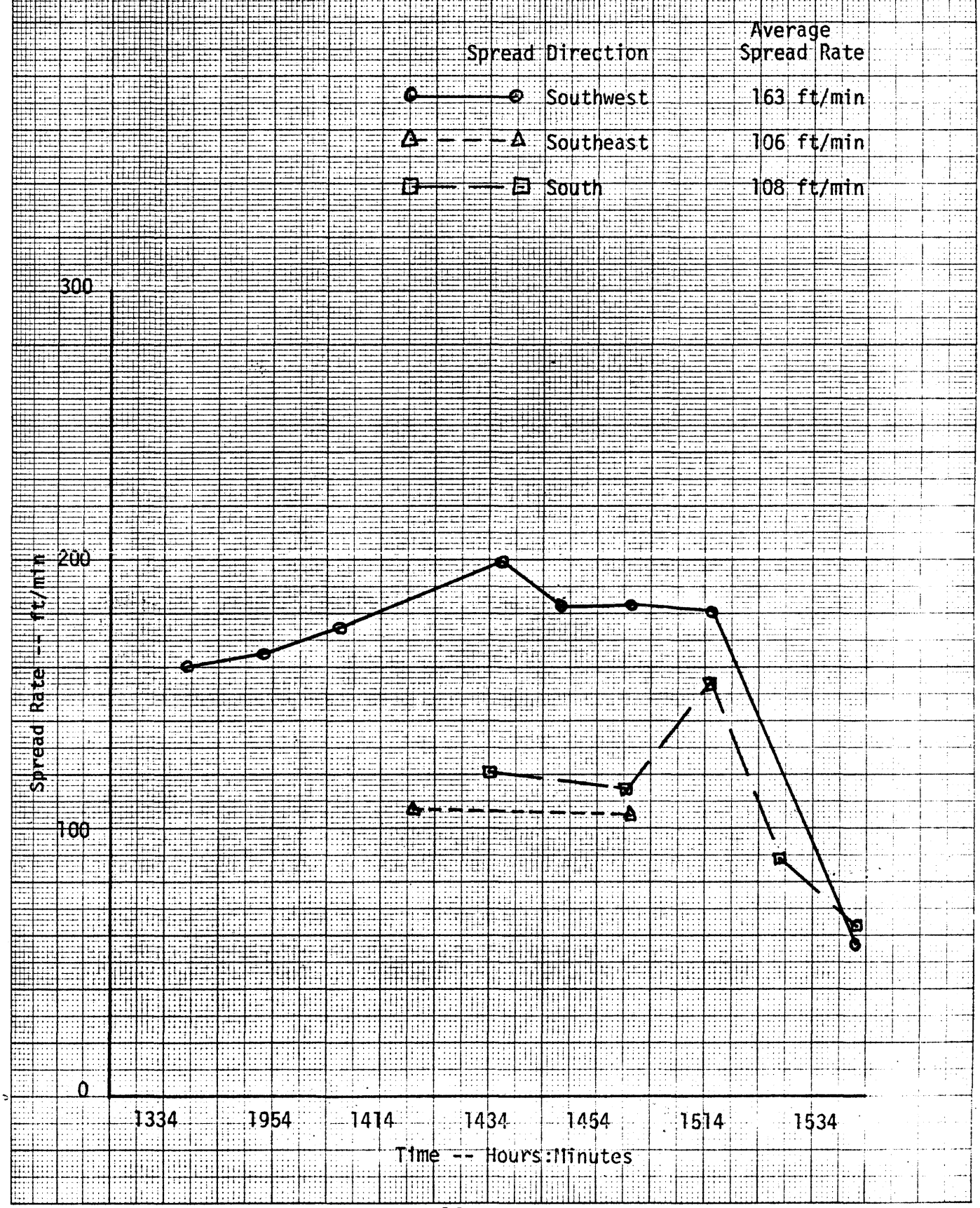
- Spread rate to the southwest is initially the same, although the measured peaks at 13:50 and 15:16 are absent from the simulated results. The simulated spread rate is about 10 percent higher between 14:14 and 15:10, thus offsetting, to some degree, the absence of the two peaks.* The average simulated spread rate over the interval is about 80 percent of the average spread rate for the measured perimters.

- Spread rate to the southeast is initially the same, rising for the measured perimeters and remaining at about the same value for the simulated perimeters. Over the interval shown, average southeast spread rate for the simulated perimeters is about 75 percent of average southeast spread rate for the measured perimeters.

- Spread rate to the south for the simulated perimeters is low during the entire interval shown, with an average value of about 60 percent of the average southward spread rate for the measured perimeters.

*Examination of the measured length shown in the upper part of Figure 8 at about 13:50 and 15:10 shows that the integrated effect of these two large measured peaks in southwest spread rate is small.

Figure 9. Rates of Spread from Simulated Potrero Fire Based on Prototype Fuel Characteristics Algorithms



- Spread rate to the southeast is initially the same, rising for the measured perimeters and remaining at about the same value for the simulated perimeters. Over the interval shown, average southeast spread rate for the simulated perimeters is about 75 percent of average southeast spread rate for the measured perimeters.

- Spread rate to the south for the simulated perimeters is low during the entire interval shown, with an average value of about 60 percent of the average southward spread rate for the measured perimeters.

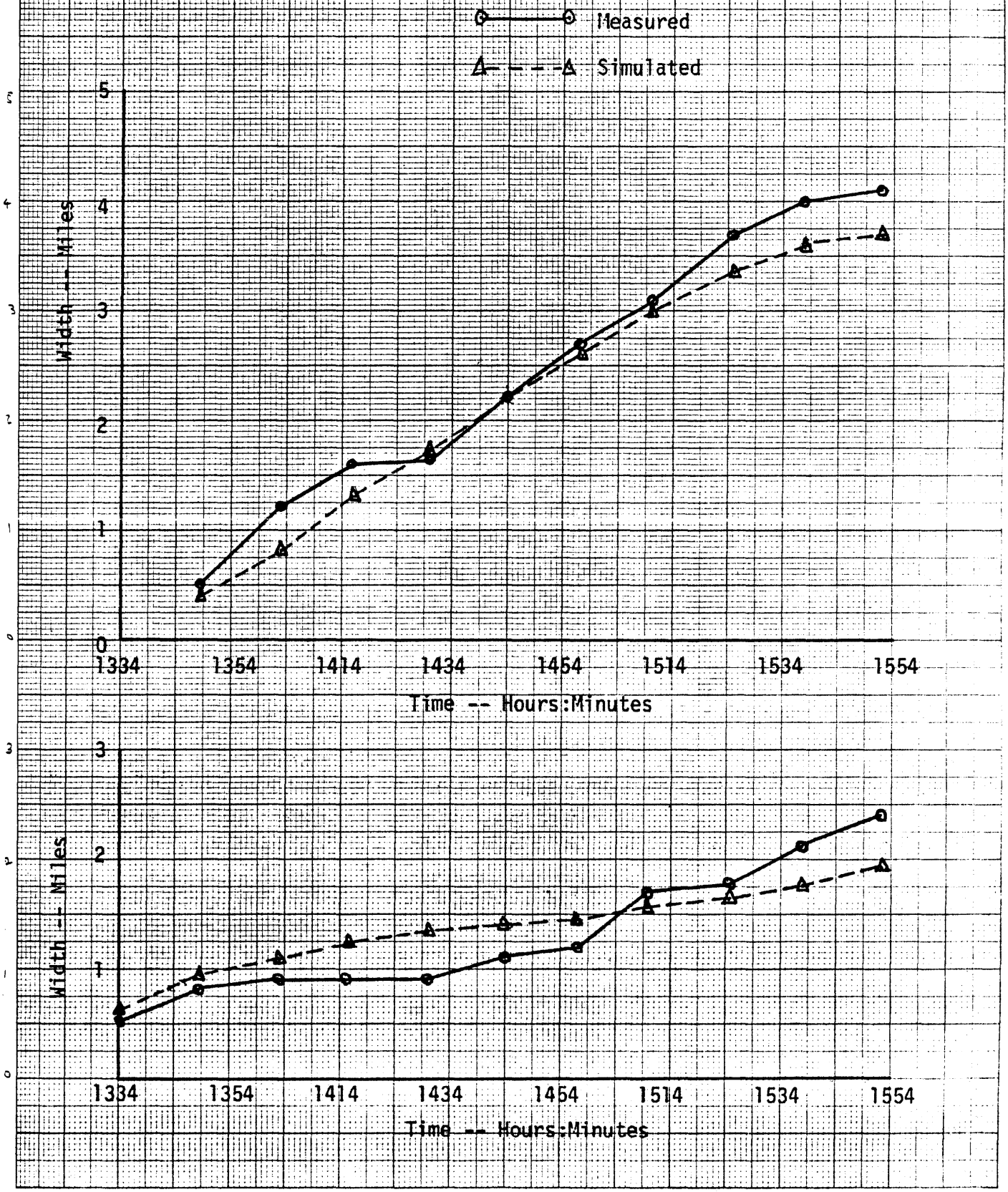
Figure 10 shows the length and width of the measured perimeters and the simulated perimeters employing fuel characteristics algorithms. The lower average, southwesterly spread rate of the simulated perimeters with fuel algorithms results in a slightly shorter cumulative length, particularly late in the fire history, relative to both the measured perimeters and the simulated perimeters based on tabulated fuel characteristics (see Figure 8). The perimeter widths in the lower part of Figure 10 show that the simulated perimeters based on fuel characteristics algorithms, like the simulated perimeters based on tabulated fuel characteristics, are wider early in the fire and narrower late in the fire, because the late southward run strongly evident in the measured perimeters is only weakly exhibited by the simulated perimeters.

Comparison of the measured perimeters, the simulated perimeters based on tabulated fuel characteristics and the simulated perimeters based on fuel characteristics algorithms shows:

- the brush fuel model produces consistently slightly lower spread rates than the tabulated sage fuel data.

- the spread rate differences between the brush model and the sage table are not sufficient to require inclusion of coastal sage as a fuel type in the fuel characteristics algorithms at the present time.

Figure 10. Length and Width of Measured Perimeters and Simulated Perimeters Based on Fuel Characteristics Algorithms



3.0 LOCAL SPREAD RATE PATTERN

The elliptical, local spread rate pattern used to generate the perimeters of Figures 1, 3 and 4 is shown in Figure 11 and is described in detail in an earlier report.[2] The choice of an ellipse, although based to some extent upon empirical data, was largely arbitrary. An alternative local spread rate pattern was examined during the current contract period.

The spread rate model defines a fuel dependent (no-slope, no-wind) spread rate R_o in terms of local fuel characteristics.[1] The objective of establishing a local spread rate pattern is to define local spread rate, in an arbitrary direction in terms of R_o (i.e., local fuel characteristics), local wind velocity, and local topography The spread rate model defines the local spread rate in the direction of the wind vector as[1]:

$$R(\alpha_w) = R_o[1 + \phi_w(S_w) + \phi_s(\theta_w)]$$

where α_w is wind azimuth angle, ϕ_w and ϕ_s are scalar functions of the wind speed (S_w) and slope (θ_w), and the wind direction is the same as the up-slope direction. The above formulation can be used to define local spread rate in an arbitrary direction in terms of local conditions as described below.

Consider a location for which R_o has been specified in terms of local fuel characteristics, where the wind velocity is $\vec{V}_w$ and the slope-aspect is $\theta\hat{a}_a$. Let it be desired to determine the spread rate in an arbitrary direction specified by the unit vector $\hat{a}_s$. Define the component of wind speed in the direction a_s as $S_w = \vec{V}_w \cdot \hat{a}_s$, and the component of slope in the direction $\hat{a}_s$ as $\theta_w = \theta\hat{a}_a \cdot \hat{a}_s$. Thus from above, the spread rate in an arbitrary direction is given by the rate of spread model in terms of the local components of the variables (hence, local component spread rate pattern) as:

$$R(\hat{a}_s) = R[1 + \phi_w(\vec{V}_w \cdot \hat{a}_s) + \phi_s(\theta\hat{a}_a \cdot \hat{a}_s)].$$

A = 5 mph
B = 10
C = 15
D = 20
E = 25
F = 30
G = 35
H = 40
I = 45
J = 50

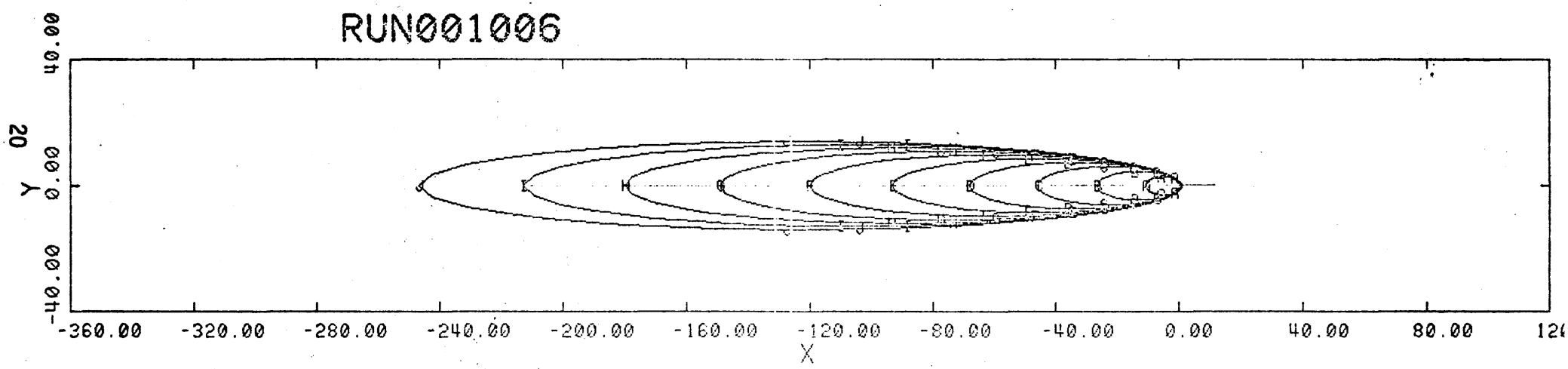


Figure 11

Elliptical, Local Spread Rate Patterns for Several Values of Wind Speed

The above relationship is plotted in Figure 12 for three values of wind speed, one wind direction, one slope-aspect and one value of R_o. The discontinuities near the origin result from limiting $R(\hat{a}_s)$ such that $R(\hat{a}_s) \geq R_o$. This limiting condition was chosen since, for wind speeds significantly greater than zero, $R_{mx} >> R_o$, and zero or negative spread rates are not considered significant.

The Potrero Fire simulation was exercised using the local spread rate pattern of Figure 12 and the fuel characteristics algorithms. The results are shown in Figure 13, where it is clear that the primary effect of the local component spread rate pattern is to broaden and to blunt the heads of the perimeters. Examination of the measured perimeters of Figure 2 shows that the actual tendency is for the heads of the perimeters to become elongated (under conditions of high wind speeds) in the direction of maximum propagation rate. This tendency is seen in Figure 2 to persist to the southwest until about 15:25, when the coast is reached. Thereafter elongation tends to occur toward the south in the new direction of maximum propagation rate. The formation of such elongated heads under high wind conditions has been found to be a characteristic of the elliptic spread pattern.[3] This, together with the results of Figure 13 led to the conclusion that the local component spread rate pattern is sufficiently inferior to the elliptic local spread rate pattern that further examination was unnecessary.

4.0 LOCATION OF FIRE ORIGIN AND REDISTRIBUTION OF PERIMETER POINTS

4.1 LOCATION OF FIRE ORIGIN

Because the initial fire location may not be accurately known, it is of interest to have some knowledge of how sensitive the final perimeters are to initial fire location. A very preliminary examination of this effect was made by moving the fire origin approximatley 120 feet to the east and running the simulation with the elliptical local spread model and fuel characteristics

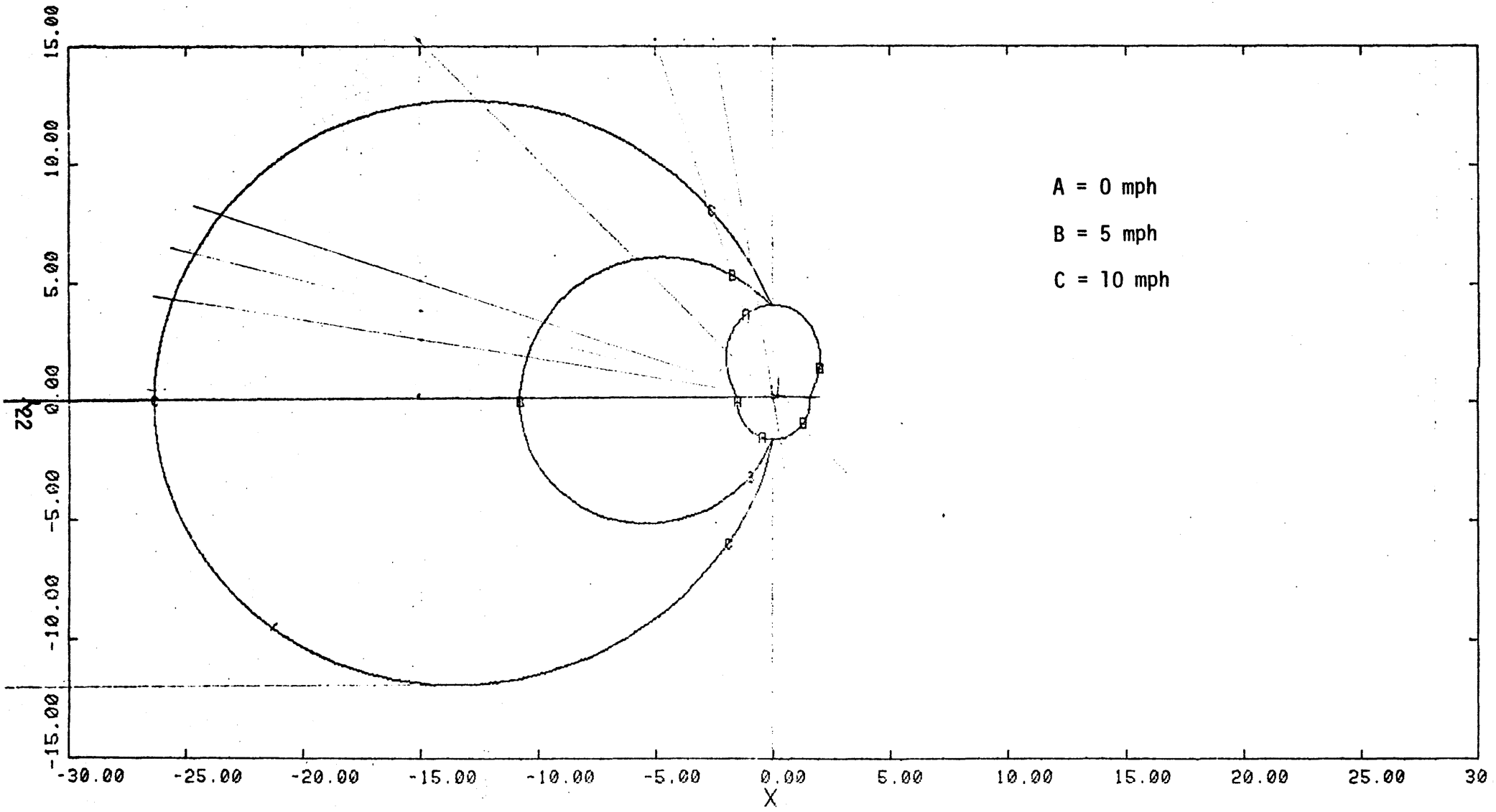


Figure 12

Local Component Spread Pattern

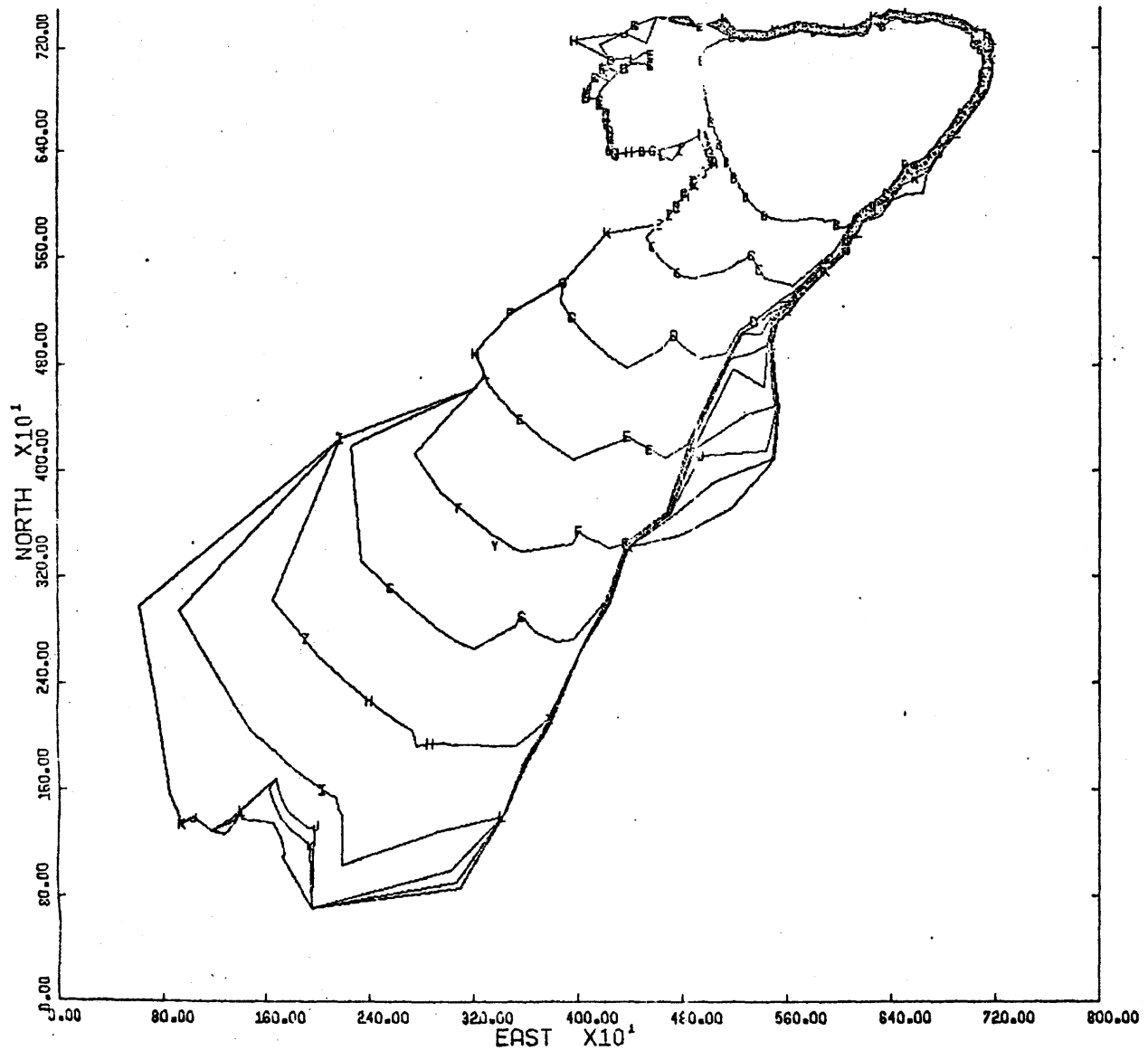


Figure 13

Potrero Fire Simulation for Local Component Spread Rate Pattern

algorithms to obtain the perimeters of Figure 14. With the exception of the altered origin, Figure 4 and Figure 14 are thus directly comparable (Note: Perimeter C of Figure 14 = perimeter B of Figure 4, etc., because of an additional initial perimeter in Figure 14). Comparison of Figure 4 and Figure 14 reveals that differences in the two sets of perimeters are mostly of second order, although there is a stronger trend for propagation to the southward exhibited by the perimeters of Figure 14. Location of the fire origin may, in the case of the Potrero fire, be much more sensitive than would normally be expected, because of the topography. The major line of propagation through 15:25 is toward the southwest (see Figure 2) along a ridge. A displacement of the origin that affects its location relative to the ridge could significantly affect later perimeters. This is probably the source of the increased trend toward southward propagation in Figure 14.

4.2 PERIMETER POINT REDISTRIBUTION

Examination of the perimeters of Figure 4 reveals, for perimeters later than 14:16, relatively long, straight-line segments at the head and/or flanks, but not at the tail. The reason for this is a failure of the perimeter point distribution model. Because the minimum and maximum resolution lengths, together with the maximum allowable number of perimeter points are initially fixed, the available perimeter points can be depleted early in the simulation and tend to propagate toward the tails of the perimeters due to increasing perimeter length, which occurs principally at the head of the fire.

The perimeter point redistribution algorithm dynamically alters the maximum and minimum resolution distances consistent with the length of the most recently calculated perimeter. Periodically a maximum of 75 percent of the allowed perimeter points are uniformly redistributed about the perimeter. The remaining 25 percent of the allowed perimeter points may be distributed by the resolution algorithm between the points exhibiting the highest spread rates. As a result, resolution is allowed to degrade gracefully with increasing perimeter length, within the limit imposed by the maximum number of

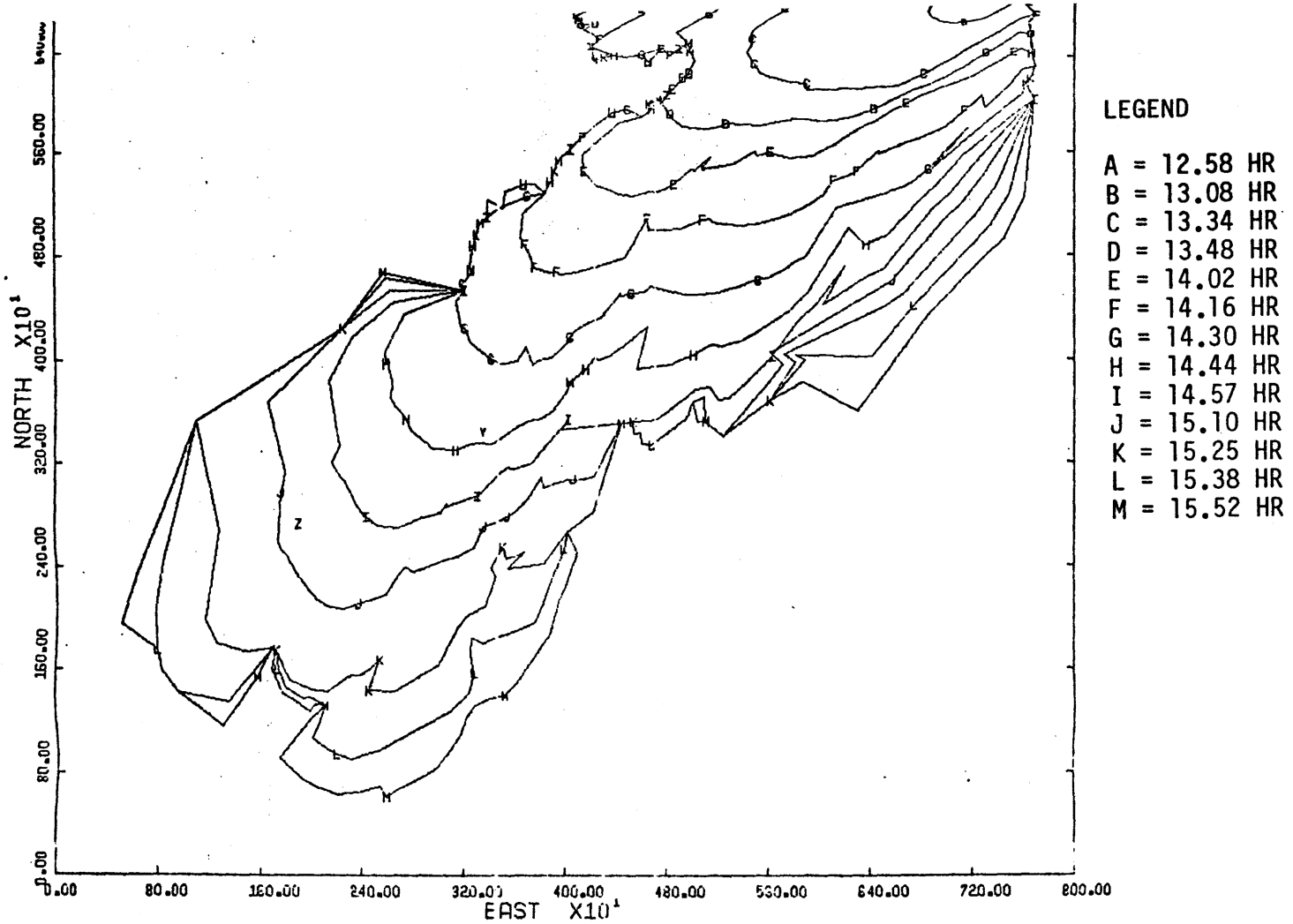


Figure 14

Potrero Fire Simulation with Origin Displaced 120 Ft. to Right

allowable perimeter points.

The Potrero fire simulation was run with the elliptical spread pattern, fuel characteristics algorithms, the perturbed origin and the dynamic perimeter point redistribution algorithm. The results are shown in Figure 15. Thus, Figure 15 differs from Figure 14 only in the use of perimeter point redistribution. Comparison of Figures 14 and 15 reveals:

- The primary effect of perimeter point redistribution is to smooth the perimeter, particularly in regions of rapid spread.

- Concavities in the perimeter that are also characterized by very low spread rates (upper center of Figure 15) may become blurred due to multiple representation by slightly different sets of perimeter points. This is not considered significant because the blurring is slight and regions of very low spread rate are not of prime importance.

- Redistribution of perimeter points provides a slightly more realistic representation of an actual fire than the propagation of a fixed set of points from perimeter to perimeter. Because of the redistribution the perimeter points propagate through the composite topographic, fuel and wind field in a much less deterministic manner, thus reducing the tendency to follow topographic contour lines or wind stream lines.

Spread rates taken from the perimeters of Figure 15 are presented in Figure 16. Comparison of Figure 16 and Figure 6 reveals:

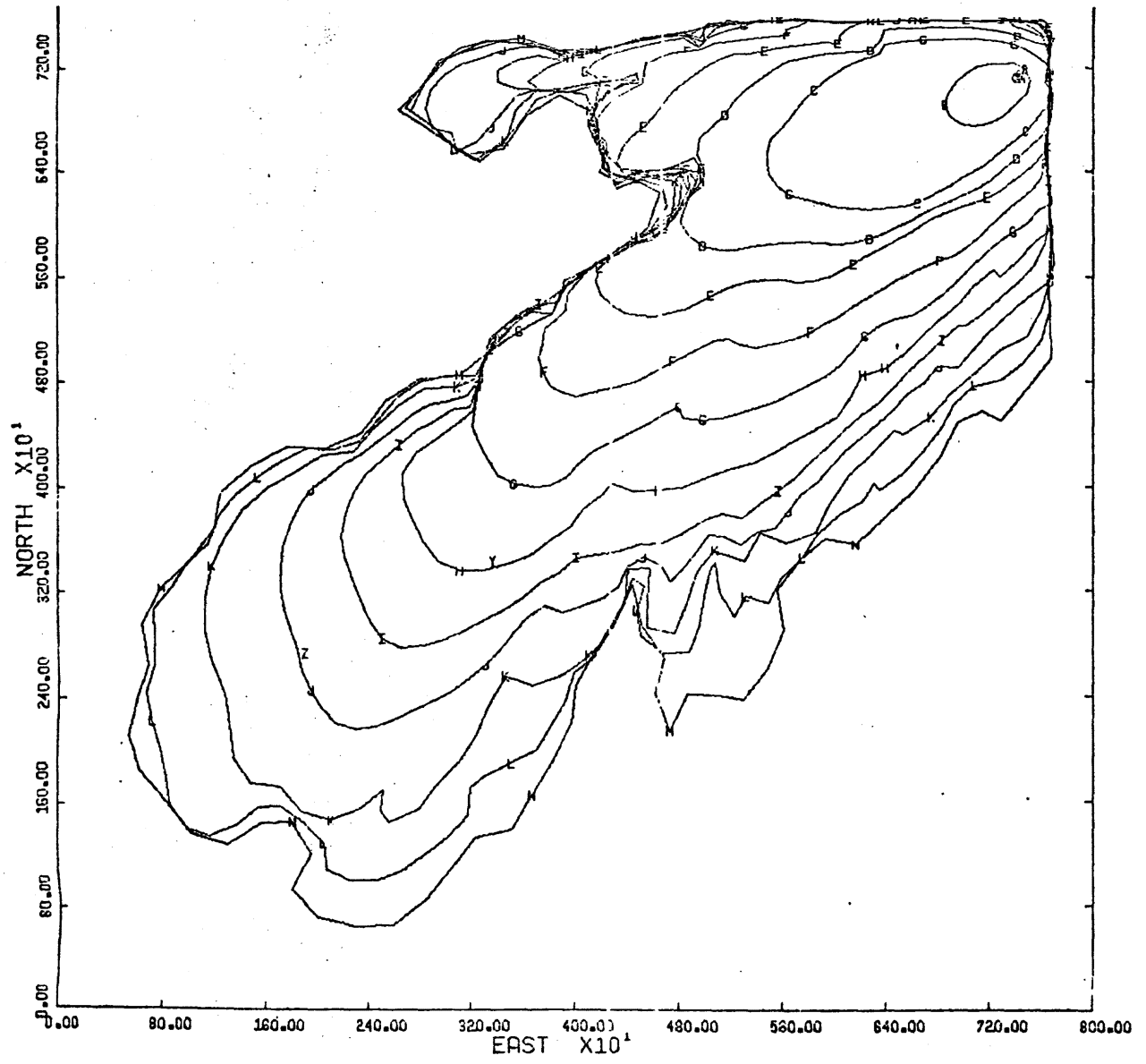
- To the southwest the calculated spread rates are consistently high, except for the two measured peaks at 13:52 and 15:16. These two peaks are, however, weakly represented in the calculated results. Over the entire interval shown in Figure 16 the average

calculated spread rate to the southwest differs from the average measured spread rate to the southwest by about 1 percent.

- To the southeast the initial calculated spread rate is high, the final value is low and the average calculated spread rate to the southeast is about 82 percent of the average measured value.

- Calculated spread rate to the south is consistently low, the large measured peak at about 15:29 is weakly represented in the calculated results and the average calculated spread rate to the south is about 77 percent of the measured value.

The lengths and widths of the Potrero fire as represented by the measured perimeters of Figure 2 and the calculated perimeters of Figure 15 are shown in Figure 17. The length curves of Figure 17 (upper) reflect the effects of the southwest spread rate curves in Figure 16. The initial lengths are about the same, but the measured length grows more rapidly from 13:38 to about 14:10 due to the first large spread rate peak of Figure 6. For times greater than 14:10 the higher average spread rate of the simulated perimeters produces consistently greater lengths, even during the second large spread rate peak of Figure 6. The relative importance of the spread rate peaks versus average value is clearly shown in Figure 17. The cumulative effects of the large measured southwest spread rate peaks at 13:52 and 15:16 (see Figure 6) appear in the measured curve of Figure 17 as minor purturbations in length. The width curves in Figure 17 show that the width of the calculated perimeters is consistently greater than the measured perimeter widths until about 15:34. The measured perimeters exhibit a rapid southward run from 15:25 to 15:52 (see Figure 2) which is only weakly evidenced by the simulated perimeters (see Figure 16). The widths of both perimeters were determined at the widest point of the measured perimeters. Because of the late southward run, the widest point of the measured perimeters shifted from the rear (upper right of Figure 2) to the front (lower left of Figure 2). This did not occur for the simulated



LEGEND

A = 12.58 HR
B = 13.08 HR
C = 13.34 HR
D = 13.48 HR
E = 14.02 HR
F = 14.16 HR
G = 14.30 HR
H = 14.44 HR
I = 14.57 HR
J = 15.10 HR
K = 15.25 HR
L = 15.38 HR
M = 15.52 HR

Figure 15

Potrero Fire Simulation With Dynamic Perimeter Point Redistribution

perimeters so that the calculated width shown in Figure 17 appears to decrease for times greater than 15:25, due to a shift in the point of measurement.

5.0 CONCLUSIONS

The current prototype segmented perimeter propagation model appears to exhibit sufficient fidelity to provide a useful procedure for simulating large Southern California wildland fires in terms of specified fuel type and age, meteorological conditions and topography. The segmented propagation model is currently programmed for a CDC 7600 computer. It has been determined that the segmented propagation model can be programmed on a PDP 11/45 mini-computer. Such a program will permit accurate determination of mini-computer storage and mini-computer running times required to simulate wildland fires of various sizes to various levels of precision.

Figure 16. Spread Rates from Simulated Potrero Fire with Perimeter Point Redistribution

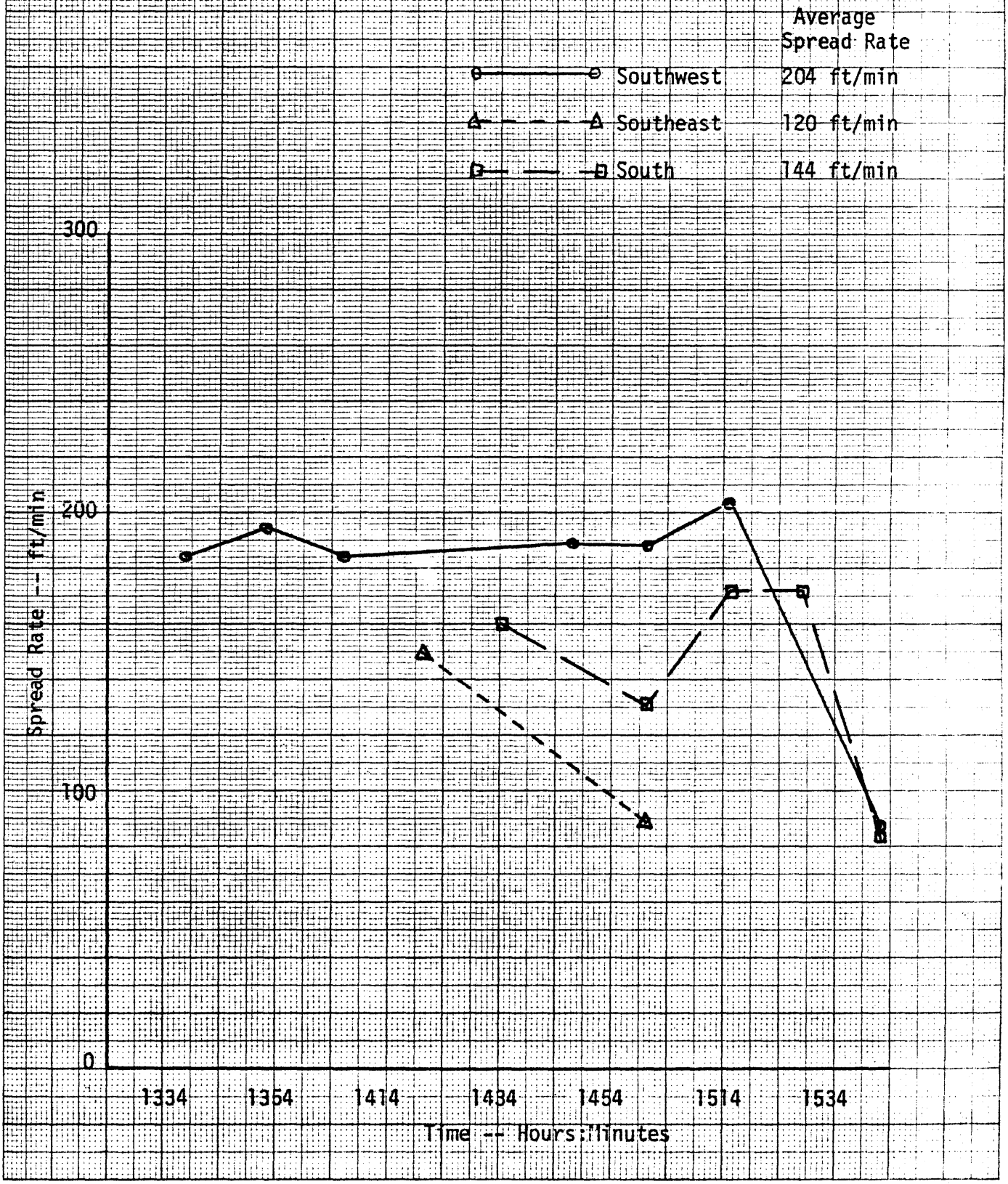
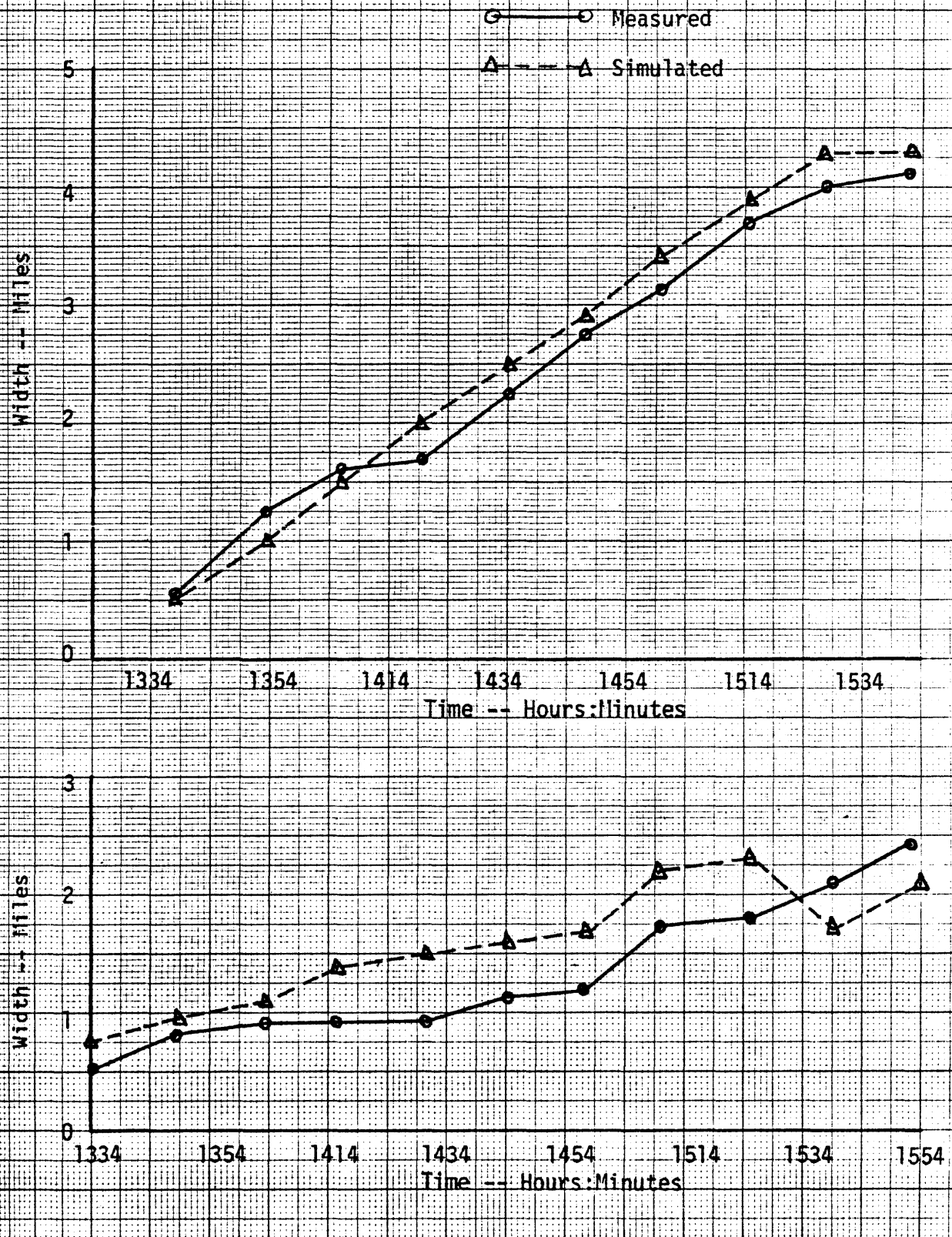


Figure 17. Length and Width of Measured Perimeters and Simulated Perimeters Using Perimeter Point Redistribution



6.0 SEGMENTED PERIMETER PROPAGATION MODEL

6.1 SYSTEM MANAGER (RADSM)

1. Purpose

This routine directs the calculation of segmented fire perimeter estimated by a radial growth concept.

2. Arguments

INPUTS

NPMX	-	The maximum number of points in any perimeter
IMAX	-	The maximum number of storage locations available for any one perimeter
NPOUT	-	The number of perimeters to be output. This includes the first one which is also output.
JMAX	-	The maximum number of perimeters which may be stored
PT	-	An array of perimeter times in minutes relative to TOF, IDOF, to be calculated, PT(1) is the time of the first perimeter, which is input. Length is NPOUT. The order must be increasing.
DTMX	-	Maximum Delta T to be used in calculating internal perimeters.
ROEØ	-	Base resolution parameter
ROEMN	-	Minimum resolution parameter
ISS	-	Scatter storage function
TOF	-	Time of fire in minutes from midnight
IDOF	-	Day of year
PTLR	-	Perimeter time in minutes from TOF (IDOF) of last real perimeter

TODMAX	-	Maximum time of day in minutes (=1440)
SAMSIZ	-	Sample interval size for propagating along a line
MXMASH	-	Maximum length of mash table
MXMCCT	-	Maximum length of MCCT table
NORM	-	Normal maximum number of cells, a prime number (normally much greater than NPMX)
MARG	-	Number of scatter storage collistion cells

NOTE: MXMASH must be greater than or equal to (NORM+MARG) x 3

INPUTS/OUTPUTS

MASH	-	Scatter storage table. Length = (NORM+MARG) x 3
MCCT	-	Massage cell collision table. Length = NORM x 2
XINT	-	Internal perimeter X coordinate linked list
YINT	-	Internal perimeter Y coordinate linked list
POLANG	-	An array of polar angles which indicate the direction a point moved on its last advance (LENGTH=IMAX)
DISP	-	An array which contains the distance a point moved on its last advance (LENGTH=IMAX)
LINKF	-	Forward link for XINT, YINT
LINKB	-	Backward link for XINT, YINT
NPTS	-	An array containing the number of points in each perimeter. Length = NPOUT. NPTS(1) = The number of points in the input perimeter.
XP	-	X coordinates of all perimeters. Dimensioned (IMAX,JMAX) the first subscript designates a particular perimeter point. The second subscript designates a particular perimeter. XP(K,1) is the Kth point on the input perimeter.
YP	-	Y coordinates of all perimeters. Analogous to XP.

NOTE: DISP and POLANG may occupy the same storage area as MCCT, but DISP and POLANG may not occupy the same storage area.

3. Procedure

The primary concerns of this routine are the location of perimeter data, the form of the data and the sequencing of calculations to generate the data. The routine is basically one loop which calculates a specified perimeter each time through the loop. Since perimeters may be specified at any times, another set of internal perimeters must be calculated at a controlled time interval to maintain accurate perimeter descriptions.

The first operation in the main loop is to copy the preceding perimeter into the sequential storage area for the next perimeter. This copy is necessary since the new sequential storage area is used as a working buffer area during the calculation of internal perimeters.

The next operation calculates the time interval for growth of the next internal perimeter. The time interval is calculated by a routine called DLTRM. Successive operations calculate the internal perimeter growth and first order point filtering (RADPER), internal perimeter smoothing (MASSAG), internal perimeter point redistribution and conversion to sequential storage form (REDIST), internal perimeter time interval recalculation (DLTRM) and recalculation of resolution parameters (RESSET). If the last internal perimeter corresponds with the next desired perimeter, the length of the perimeter is calculated and the next pass through the main loop begins; otherwise, the operations are repeated beginning with the internal perimeter growth calculation (RADPER).

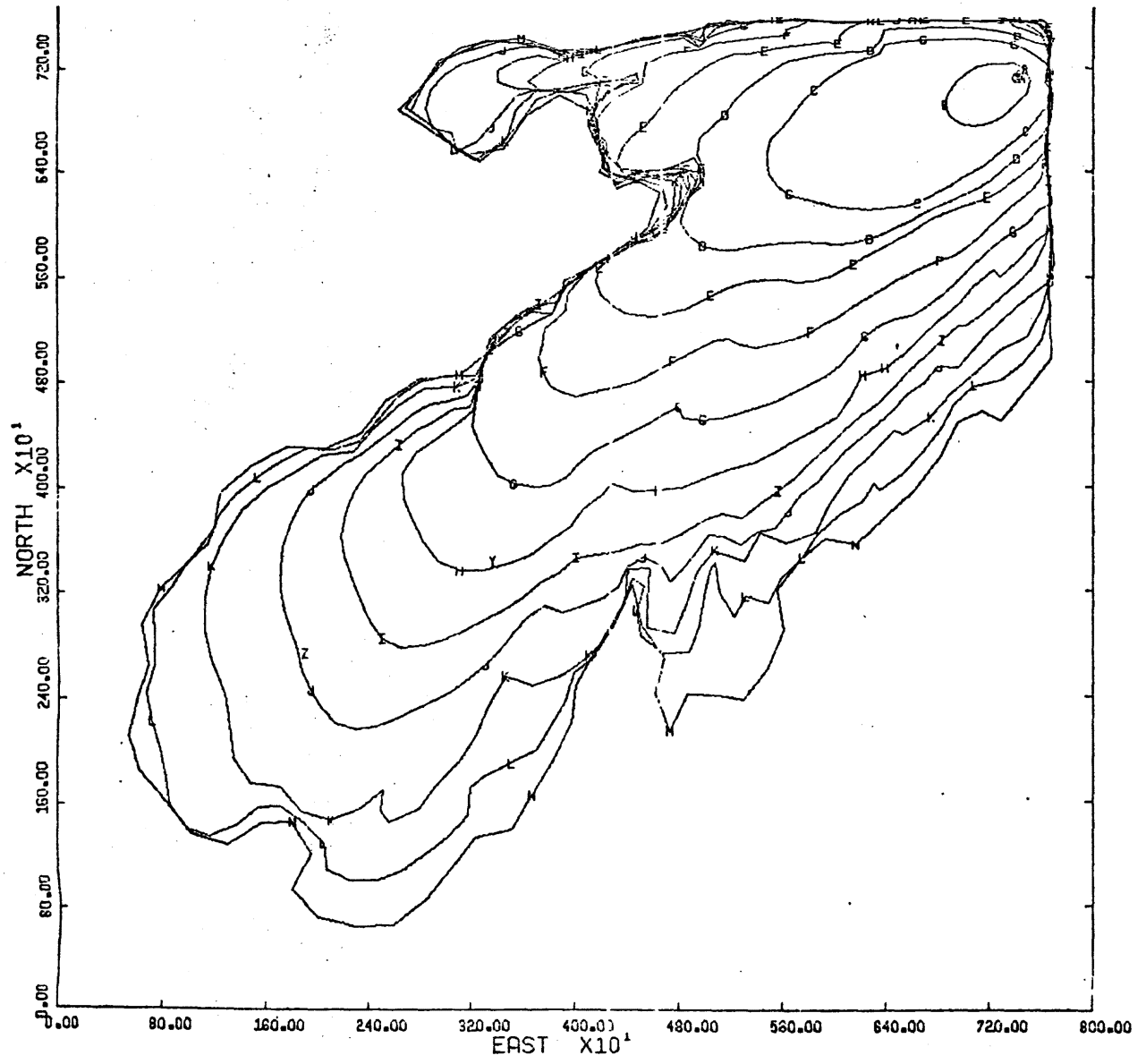
4. Comments

The above procedure may calculate any number of perimeters for which there is storage. In addition, the procedure may be restarted from any perimeter which it generates.

The figure on the following page shows the hierarchy of Radial Spread Model Routines. This figure is included to provide an overview of the structure which RADSM controls.

5. Flow Chart

See Figure 19.



LEGEND

A = 12.58 HR
B = 13.08 HR
C = 13.34 HR
D = 13.48 HR
E = 14.02 HR
F = 14.16 HR
G = 14.30 HR
H = 14.44 HR
I = 14.57 HR
J = 15.10 HR
K = 15.25 HR
L = 15.38 HR
M = 15.52 HR

Figure 15

Potrero Fire Simulation With Dynamic Perimeter Point Redistribution

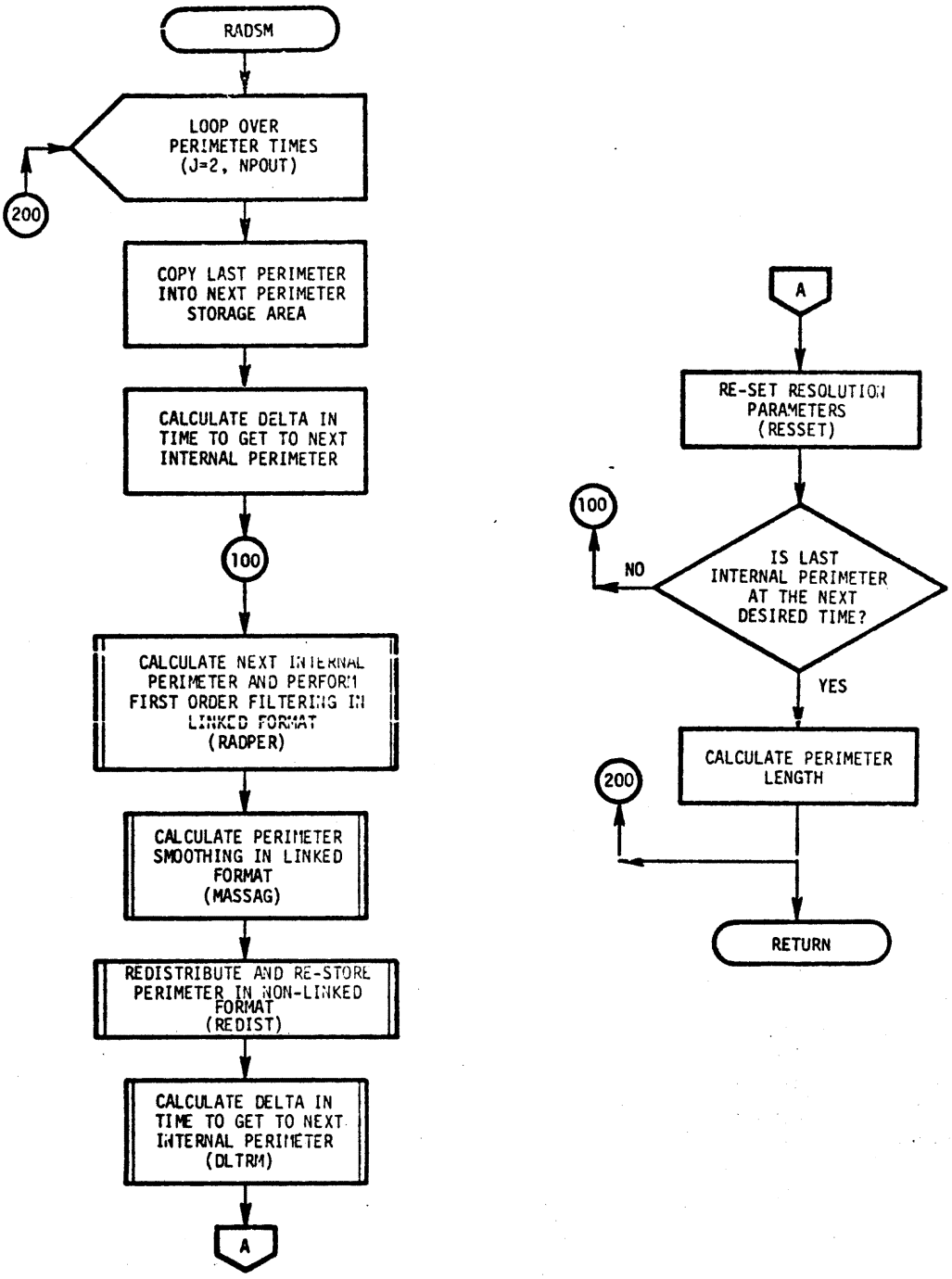

RADSM
LOOP OVER PERIMETER TIMES (J=2, NPOUT)
200
COPY LAST PERIMETER INTO NEXT PERIMETER STORAGE AREA
CALCULATE DELTA IN TIME TO GET TO NEXT INTERNAL PERIMETER
100
CALCULATE NEXT INTERNAL PERIMETER AND PERFORM FIRST ORDER FILTERING IN LINKED FORMAT (RADPER)
CALCULATE PERIMETER SMOOTHING IN LINKED FORMAT (MASSAG)
REDISTRIBUTE AND RE-STORE PERIMETER IN NON-LINKED FORMAT (REDIST)
CALCULATE DELTA IN TIME TO GET TO NEXT INTERNAL PERIMETER (DLTRM)
A
A
RE-SET RESOLUTION PARAMETERS (RESSET)
100
NO
IS LAST INTERNAL PERIMETER AT THE NEXT DESIRED TIME?
YES
200
CALCULATE PERIMETER LENGTH
RETURN

Figure 19

Radial System Manager

```
      SUBROUTINE RADSM(NPMX,IMAX,NPOUT,JMAX,PT,DTMX,ROE0,ROEMN,ISS,
     + TOF,IDOF,PTLR,TODMAX,SAMSIZ,MXMASH,MXMCCT,NORM,MARG,MASH,MCCT,
     + XINT,YINT,POLANG,DISP,LINKF,LINKB,NPTS,XP,YP)
C
C     THIS ROUTINE CALCULATES FIRE PERIMETERS AS RADIALLY GROWING SIMPL
C     CLOSED BOUNDARIES.
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     NPMX    - THE MAXIMUM NUMBER OF POINTS IN ANY PERIMETER
C     IMAX    - THE MAXIMUM NUMBER OF STORAGE LOCATIONS AVAILABLE FOR AN
C               ONE  PERIMETER
C     NPOUT   - THE NUMBER OF PERIMETERS TO BE OUTPUT.  THIS INCLUDES TH
C               FIRST ONE WHICH IS ALSO OUTPUT.
C     JMAX    - THE MAXIMUM NUMBER OF PERIMETERS WHICH MAY BE STORED
C     PT      - AN ARRAY OF PERIMETER TIMES IN MINUTES  RELATIVE TO
C               TOF,IDOF, TO BE CALCULATED, PT(1) IS THE TIME OF THE
C               FIRST PERIMETER, WHICH IS INPUT.  LENGTH = NPOUT.
C               THE ORDER MUST BE INCREASING.
C               PERIMETERS.
C     ROE0    - BASE RESOLUTION PARAMETER
C     ROEMN   - MINIMUM RESOLUTION PARAMETER
C     ISS     - SCATTER STORAGE FUNCTION
C     TOF     - TIME OF FIRE IN MINUTES FROM MIDNIGHT
C     IDOF    - DAY OF YEAR
C     PTLR    - PERIMETER TIME IN MINUTES FROM TOF(IDOF) OF LAST REAL
C               PERIMETER
C     TODMAX  - MAXIMUM TIME OF DAY IN MINUTES (=1440)
C     SAMSIZ  - SAMPLE INTERVAL SIZE FOR PROPAGATING ALONG A LINE
C     MXMASH  - MAXIMUM LENGTH OF MASH TABLE
C     MXMCCT  - MAXIMUM LENGTH OF MCCT TABLE
C     NORM    - NORMAL MAXIMUM NUMBER OF CELLS, A PRIME NUMBER
C               (NORMALLY MUCH GREATER THAN NPMX)
C     MARG    - NUMBER OF SCATTER STORAGE COLLISTION CELLS
C  NOTE..... MXMASH MUST BE .GE. (NORM+MARG)*3
C         INPUTS/OUTPUTS
C     MASH    - SCATTER STORAGE TABLE. LENGTH = (NORM+MARG)*3
C     MCCT    - MASSAGE CELL COLLISION TABLE.   LENGTH = NORM*2
C     XINT    -
C     YINT    -
C     POLANG  -
C     DISP    -
C     LINKF   -
C     LINKB   -
C     NPTS    - AN ARRAY CONTAINNING THE NUMBER OF POINTS IN EACH
C               PERIMETER.  LENGTH = NPOUT.  NPTS(1)= THE NUMBER OF
C               POINTS IN THE INPUT PERIMETER.
C     XP      - X COORDINATES OF ALL PERIMETERS.  DIMENSIONED( IMAX,JMAX
C               THE FIRST SUBSCRIPT DESIGNATES A PARTICULAR PERIMETER
C               POINT.  THE SECOND SUBSCRIPT DESIGNATES A PARTICULAR
C               PERIMETER.  XP(K,1) IS THE KTH POINT ON THE INPUT
C               PERIMETER.
C     YP      - Y COORDINATES OF ALL PERIMETERS. ANALOGUS TO XP.
C
      COMMON /NEXTRS/ RA0,RA1,RA2,RFUNC(21),INTERL
C
      EXTERNAL ISS
      DIMENSION NPTS(NPOUT),PT(NPOUT)
      DIMENSION XP(IMAX,JMAX),YP(IMAX,JMAX)
      DIMENSION POLANG(IMAX),DISP(IMAX)
      DIMENSION XINT(IMAX),YINT(IMAX)
      DIMENSION LINKF(IMAX),LINKB(IMAX)
      DIMENSION MASH(MXMASH),MCCT(MXMCCT)
      DIMENSION IRAY(7)
      DATA RFUNC/1.0,.995,.99,.985,.98,.975,.97,.965,.96,.955,
     + .95,.94,.93,.92,.91,.90,.875,.85,.65,.35,0.0/
```

```
C
C
C     LOOP OVER PERIMETERS TO BE CALCULATED
C
      INTERL=2
      DO 200 J=2,NPOUT
      JM1=J-1
C         GET NUMBER OF POINTS IN LAST PERIMETER
      NNPTS=NPTS(JM1)
C  CAUTION.....  NNPTS MUST BE GREATER THAN OR EQUAL TO 3
C         COPY LAST PERIMETER INTO NEW PERIMETER SLOT
      DO 10 I=1,NNPTS
      XP(I,J)=XP(I,JM1)
      YP(I,J)=YP(I,JM1)
   10 CONTINUE
C         GET PERIMETER TIME
      PTLAST=PT(JM1)
C  CAUTION ... PTLAST MUST BE GREATER THAN OR EQUAL TO PTLR
C         CALCULATE DELTA T TO NEXT INTERNAL PERIMETER
      DT=DLTRM(TOF,IDOF,PTLR,PTLAST,DTMX)
      IF(PT(J).LE.PTLAST+DT)GO TO 150
C
C     CALCULATE A NEW INTERNAL PERIMETER IN A LINKED LIST FORM AND
C     ELIMINATE FIRST ORDER CROSSINGS
C
  100 CALL RADPER(TOF,IDOF,PTLAST,TODMAX,SAMSIZ,XP,YP,IMAX,NPMX,
     + JMAX,J,DT,ROSMLP,ROSMNL,ROE0,ROEMN,NNPTS,DISP,POLANG,IAVAIL,
     + IACTIV,XINT,YINT,LINKF,LINKB,ROSMCP,ROSMNC)
C         MASSAGE THE RESULT TO ELIMINATE ALL CROSSINGS
      CALL MASSAG(MXMASH,MXMCCT,ISS,IMAX,NORM,MARG,MASH,MCCT,XINT,YINT,
     + LINKF,LINKB,IACTIV,IAVAIL,NNPTS)
C         MOVE INTERNAL PERIMETER TO PERMANENT STORAGE FOR
C         NEXT CALCULATION
      CALL REDIST(NPMX,J,LINKF,IACTIV,IMAX,JMAX,
     + XINT,YINT,XP,YP,PERLEN,NNPTS,ROE0,ROEMN)
      IF(MXMASH.LT.0)GO TO 300
      PTLAST=PTLAST+DT
C         CALCULATE DELTA T TO NEXT INTERNAL PERIMETER
      DT=DLTRM(TOF,IDOF,PTLR,PTLAST,DTMX)
C         RECALCULATE RESOLUTION PARAMETERS
      ROSMLP=ROSMCP
      ROSMNL=ROSMNC
      CALL RESSET(ROE0,ROEMN,ROSMLP,ROSMNL)
C         IF NEXT OUTPUT PERIMETER DOES NOT LIE WITHIN THE INTERVAL,
C         CALCULATE ANOTHER INTERNAL PERIMETER
      IF(PT(J).GT.PTLAST+DT)GO TO 100
C         CALCULATE THE NEXT PERIMETER TO BE OUTPUT
  150 DT=PT(J)-PTLAST
      CALL RADPER(TOF,IDOF,PTLAST,TODMAX,SAMSIZ,XP,YP,IMAX,NPMX,
     + JMAX,J,DT,ROSMLP,ROSMNL,ROE0,ROEMN,NNPTS,DISP,POLANG,IAVAIL,
     + IACTIV,XINT,YINT,LINKF,LINKB,ROSMCP,ROSMNC)
      CALL MASSAG(MXMASH,MXMCCT,ISS,IMAX,NORM,MARG,MASH,MCCT,XINT,YINT,
     + LINKF,LINKB,IACTIV,IAVAIL,NNPTS)
C
C     MOVE INTERNAL PERIMETER TO PERMANENT STORAGE FOR NEXT CALCULATION
C
      CALL REDIST(NPMX,J,LINKF,IACTIV,IMAX,JMAX,
     + XINT,YINT,XP,YP,PERLEN,NNPTS,ROE0,ROEMN)
      NPTS(J)=NNPTS
      ROSMLP=ROSMCP
      ROSMNL=ROSMNC
      WRITE(6,3000)PERLEN
 3000 FORMAT(2X,24HTOTAL PERIMETER LENGTH = ,1P2E16.5)
      CALL RESSET(ROE0,ROEMN,ROSMLP,ROSMNL)
      CALL STATUS(IRAY)
      WRITE(6,8000)IRAY(1),IRAY(6),J,NNPTS
 8000 FORMAT(2X,14HELAPSED CP TIM,I20,2X,3HCUS,I20,2X,8HPER. NO. I5,2X,
     + 9HCONTAINS ,I5,1X,6HPOINTS)
      IF(MXMASH.LT.0)GO TO 300
      IF(IRAY(6).GE.6400000)GO TO 300
  200 CONTINUE
  300 NPOUT=JM1+1
      MXMASH=IABS(MXMASH)
      RETURN
      END
      LIST
```

6.2 PERIMETER POINT REDISTRIBUTION (REDIST)

1. Purpose

This routine redistributes the points on a calculated fire perimeter as required.

2. Arguments

INPUTS

NPMX	-	The maximum number of points in any perimeter
J	-	Perimeter index
LINKF	-	Forward Link
IACTIV	-	Index to the active internal perimeter points linked list
IMAX	-	The maximum number of storage locations available for any one perimeter.
JMAX	-	The maximum number of perimeters which may be stored
XINT	-	X coordinates for the internal perimeter (a linked list)
YINT	-	Y coordinates for the internal perimeter (a linked list)

INPUT/OUTPUTS

PERLEN	-	Perimeter Length
NNPTS	-	New number of points
RØEO	-	Base resolution
RØEMN	-	Minimum resolution

OUTPUTS

XP	-	X coordinates of perimeters (a sequential list)
YP	-	Y coordinates of perimeters (a sequential list)

3. Procedure

The length of the perimeter is first calculated. A nominal perimeter length is then calculated from:

$$NOMPL = NPMX \cdot (RØEO + RØEMN) \cdot 0.5.$$

Expressed in words, the nominal perimeter length is the desired average segment length times the maximum possible number of perimeter points. An adjustment factor is then calculated by:

$$ADJFCT = AMAX1\ (1.0,\ PERLEN/NØMPL)$$

In the above equation, if PERLEN is greater than NOMPL, the value of NPMX or the value of RØEO and RØEMN are too small. The adjustment factor is applied to KØEO and RØEMN to correct the problem and the total number of points is reduced to permit point insertion in areas of high spread rate. Points are then uniformly redistributed on the perimeter.

4. Comments

None.

5. Flow Chart

See Figure 20.

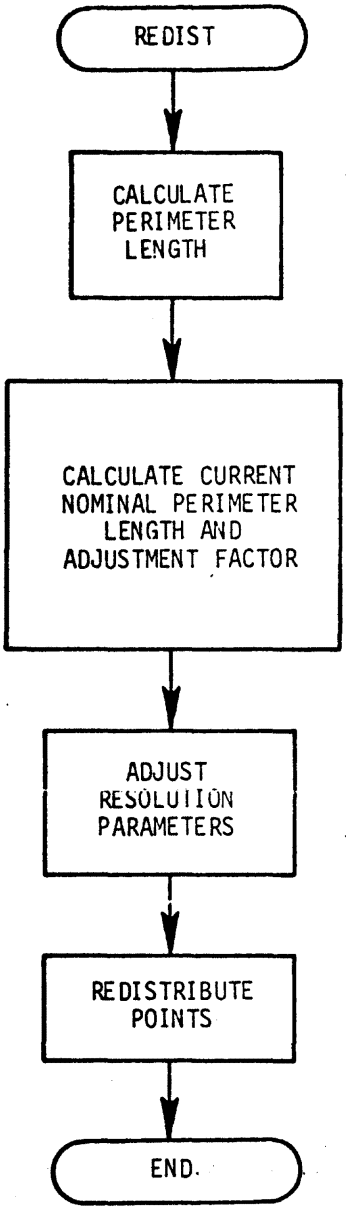


Figure 20

Perimeter Point Redistribution

```
      SUBROUTINE REDIST(NPMX,J,LINKF,IACTIV,IMAX,JMAX,
     + XINT,YINT,XP,YP,PERLEN,NNPTS,ROE0,ROEMN)
C
C     POINT AND RESOLUTION REDISTRIBUTION ROUTINE
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C     REAL NOMPL
C
      DIMENSION LINKF(IMAX),XINT(IMAX),YINT(IMAX),XP(IMAX,JMAX),
     +          YP(IMAX,JMAX)
C
C     CALCULATE LENGTH OF PERIMETER
C
      IL=IACTIV
      XLP =XINT(IL)
      YLP =YINT(IL)
      PERLEN=0.0
C
      DO 100 I=1,NNPTS
      IL=LINKF(IL)
      XL  =XINT(IL)
      YL  =YINT(IL)
      PERLEN=PERLEN+SQRT( (XLP-XL)**2 + (YLP-YL)**2 )
      XLP=XL
      YLP=YL
  100 CONTINUE
C
C     CALCULATE CURRENT NOMINAL PERIMETER LENGTH AND ADJUSTMENT FACTOR
C
      NOMPL=FLOAT(NPMX)*(ROE0+ROEMN)*0.5
      ADJFCT=AMAX1(1.0,PERLEN/NOMPL)
C
C     ADJUST RESOLUTION PARAMETERS
C
      ROE00=ROE0
      ROEMN0=ROEMN
      ROE0 = ROE0 *ADJFCT
      ROEMN= ROEMN*ADJFCT
C
C     CALCULATE REDISTRIBUTION PARAMETERS
C
      NNPTS0=NNPTS
      NNPTS = IFIX(FLOAT(NNPTS)*0.75)
      DELTAP=PERLEN/FLOAT(NNPTS)
C
C     REDISTRIBUTE POINTS
C
      IL=IACTIV
      XP(1,J)=XINT(IL)
      YP(1,J)=YINT(IL)
      XLP=XP(1,J)
      YLP=YP(1,J)
      DELTA=DELTAP
      NEWI=2
      DO 200 I=1,NNPTS0
      IL=LINKF(IL)
      XL=XINT(IL)
      YL=YINT(IL)
      PDIST= SQRT ( (XL-XLP)**2 + (YL-YLP)**2 )
  125 IF(PDIST.GE.DELTA)GO TO 150
      DELTA=DELTA-PDIST
      GO TO 199
  150 IF( PDIST .LE. 1.0E-6 ) GO TO 175
      ANG=ATAN2(YL-YLP,XL-XLP)
      XLP=COS(ANG)*DELTA+XLP
      YLP=SIN(ANG)*DELTA+YLP
      XP(NEWI,J)=XLP
      YP(NEWI,J)=YLP
      NEWI=NEWI+1
C
      PDIST=PDIST-DELTA
      DELTA=DELTAP
```

```
      GO TO 125
  175 XP(NEWI,J)=XL
      YP(NEWI,J)=YL
      NEWI=NEWI+1
      DELTA=DELTAP
  199 XLP=XL
      YLP=YL
  200 CONTINUE
      NNPTS=NEWI-1
C
      WRITE(6,1000)NNPTSO,ROEMNO,ROEØO,NNPTS,ROEMN,ROEØ,PERLEN,J
 1000 FORMAT(3X,19HREDISTRIBUTION OLD ,I3,2E13.5,5H NEW ,I3,2E13.5,
     +       4H PL ,E13.5,4H PN ,I3)
C
      RETURN
      END
```

6.3 PERIMETER POINT ADVANCE (PPTADV)

1. Purpose

This routine sequences the calculation involved in advancing a point on the estimated fire perimeter over a specified time interval.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

The direction radially outward from the fire interior is calculated. The direction of maximum spread rate is calculated and the angle of propagation is determined. The point is then advanced in small fixed distance steps until the specified time interval is reached.

4. Comments

None.

5. Flow Chart

See Figure 21.

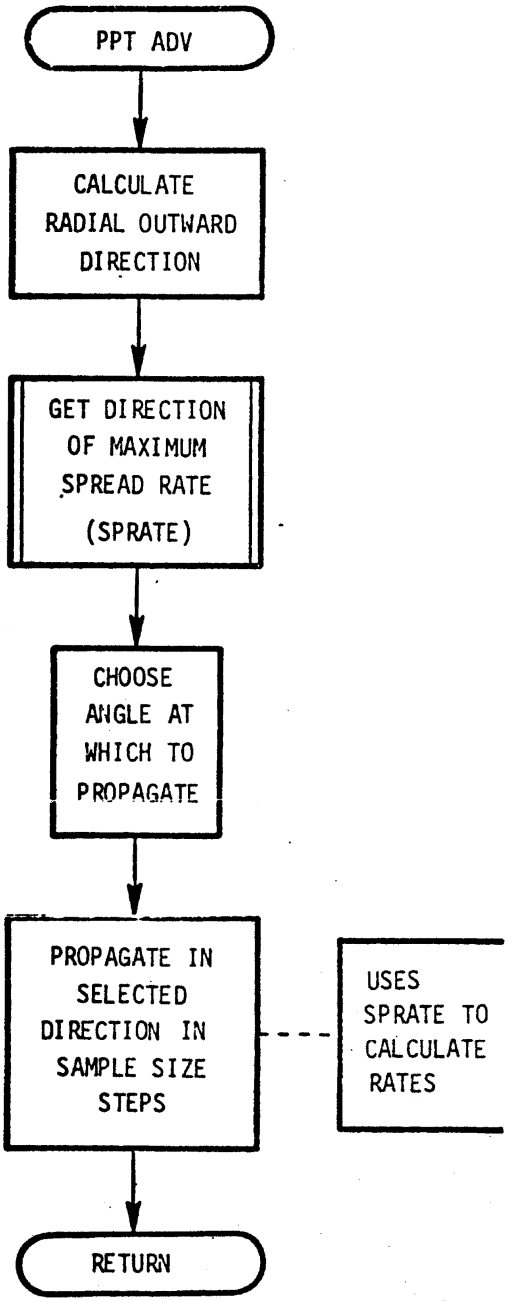

PPT ADV
CALCULATE RADIAL OUTWARD DIRECTION
GET DIRECTION OF MAXIMUM SPREAD RATE (SPRATE)
CHOOSE ANGLE AT WHICH TO PROPAGATE
PROPAGATE IN SELECTED DIRECTION IN SAMPLE SIZE STEPS
USES SPRATE TO CALCULATE RATES
RETURN

Figure 21

Perimeter Point Advances

```
      SUBROUTINE PPTADV(DT,SAMSIZ,TOPP,IDOPP,XII,YII,BETA,ALPHA,
     + ANG,XOUT,YOUT,ROSMAX,ROSMIN)
C
C     THIS ROUTINE ADVANCES A PERIMETER POINT BY TIME DT AND CALCULATES
C     A MAXIMUM AND MINIMUM RATE OF SPREAD FOUND IN THE INTERVAL.
C     VER 3.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     DT     - TIME INCREMENT (MINUTES)
C     SAMSIZ - SAMPLE INTERVAL SIZE (METERS)
C     TOPP   - TIME OF DAY OF POINT TO BE PROPAGATED
C     IDOPP  - DAY OF YEAR OF POINT TO BE PROPAGATED
C     XII    - X COORDINATE OF POINT TO BE PROPAGATED
C     YII    - Y COORDINATE OF POINT TO BE PROPAGATED
C     BETA   - POSITIVE ANGLE OF FORWARD SEGMENT
C         INPUT/OUTPUT
C     ALPHA  - POSITIVE ANGLE OF REVERSE SEGMENT
C         OUTPUTS
C     ANG    - DIRECTION OF PROPAGATION (POLAR ANGLE)
C     XOUT   - X COORDINATE OF FINAL POINT
C     YOUT   - Y COORDINATE OF FINAL POINT
C     ROSMAX - MAXIMUM RATE OF SPREAD ALONG PATH
C     ROSMIN - MINIMUM RATE OF SPREAD ALONG PATH
C
C  CAUTION  ......  IF DT IS NOT GREATER THAN 0.0
C                   THIS ROUTINE DOES NOT WORK PROPERLY
C
C         ISOLATE INPUTS
      XI=XII
      YI=YII
      DELT=DT
C         INITIALIZE MAX AND MIN OUTPUTS
      ROSMAX=0.0
      ROSMIN=1.0E30
C         INSURE ALPHA GREATER THAN BETA
      IF(ALPHA.LT.BETA)ALPHA=ALPHA+6.2831853
C         CALCULATE INITIAL PROPAGATION DIRECTION TO BE NORMAL OUTWARD
      ANG=(ALPHA-BETA)+0.5+BETA
      ALPHAN=ANG
C         CALCULATE ACTUAL PROPAGATION DIRECTION
      CALL SPRATE(XI,YI,TOPP,IDOPP,1.5707963-ANG,ROSANG,
     + ROSMX,ANGMAX)
C         CONVERT TO POLAR
      ANGMX=1.5707963-ANGMAX
C         INSURE POSITIVE
      IF(ANGMX.LT.0.0)ANGMX=ANGMX+6.2831853
C         INSURE ANGMX GREATER THAN BETA
      IF(ANGMX.LT.BETA)ANGMX=ANGMX+6.2831853
```

```
C         CHOOSE CORRECT ANGLE TO USE
      IF(ANGMX.LT.ALPHA.AND.ANGMX.GT.BETA)
     + ANG=0.5*ABS(ANGMX-ALPHAN)+AMIN1(ANGMX,ALPHAN)
      IF((ANGMX.GT.ALPHAN.AND.(ANGMX-ALPHAN).GE.1.5707963).OR.
     + (ALPHAN.GT.ANGMX.AND.(ALPHAN-ANGMX).GE.1.5707   ))ANG=ALPHAN
C         CALCULATE STEP SIZE
      COSANG=COS(ANG)
      SINANG=SIN(ANG)
      DX=SAMSIZ*COSANG
      DY=SAMSIZ*SINANG
C         CONVERT TO AZIMUTHAL ANGLE
      ANGP=1.5707963-ANG
C
C     PROPAGATE
C
    5 CALL SPRATE(XI,YI,TOPP,IDOPP,ANGP,ROSANG,ROSMX,ANGMAX)
      ROSMAX=AMAX1(ROSMAX,ROSANG)
      ROSMIN=AMIN1(ROSMIN,ROSANG+0.99999)
      DD=ROSANG*DELT
      IF(DD.LE.SAMSIZ)GO TO 10
      DELT=DELT-SAMSIZ/ROSANG
      XI=XI+DX
      YI=YI+DY
      GO TO 5
   10 XOUT=XI+DD*COSANG
      YOUT=YI+DD*SINANG
      RETURN
      END
```

6.4 PERIMETER GROWTH (RADPER)

1. Purpose

This routine calculates the growth of an internal perimeter and performs first order point filtering on the new perimeter.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

This routine loops over each point in the last perimeter and calculates the new position using PPTADV. Each segment defined is examined to see if more points are required, and points are added as necessary to satisfy the resolution specified.

4. Comments

None.

5. Flow Chart

See Figure 22.

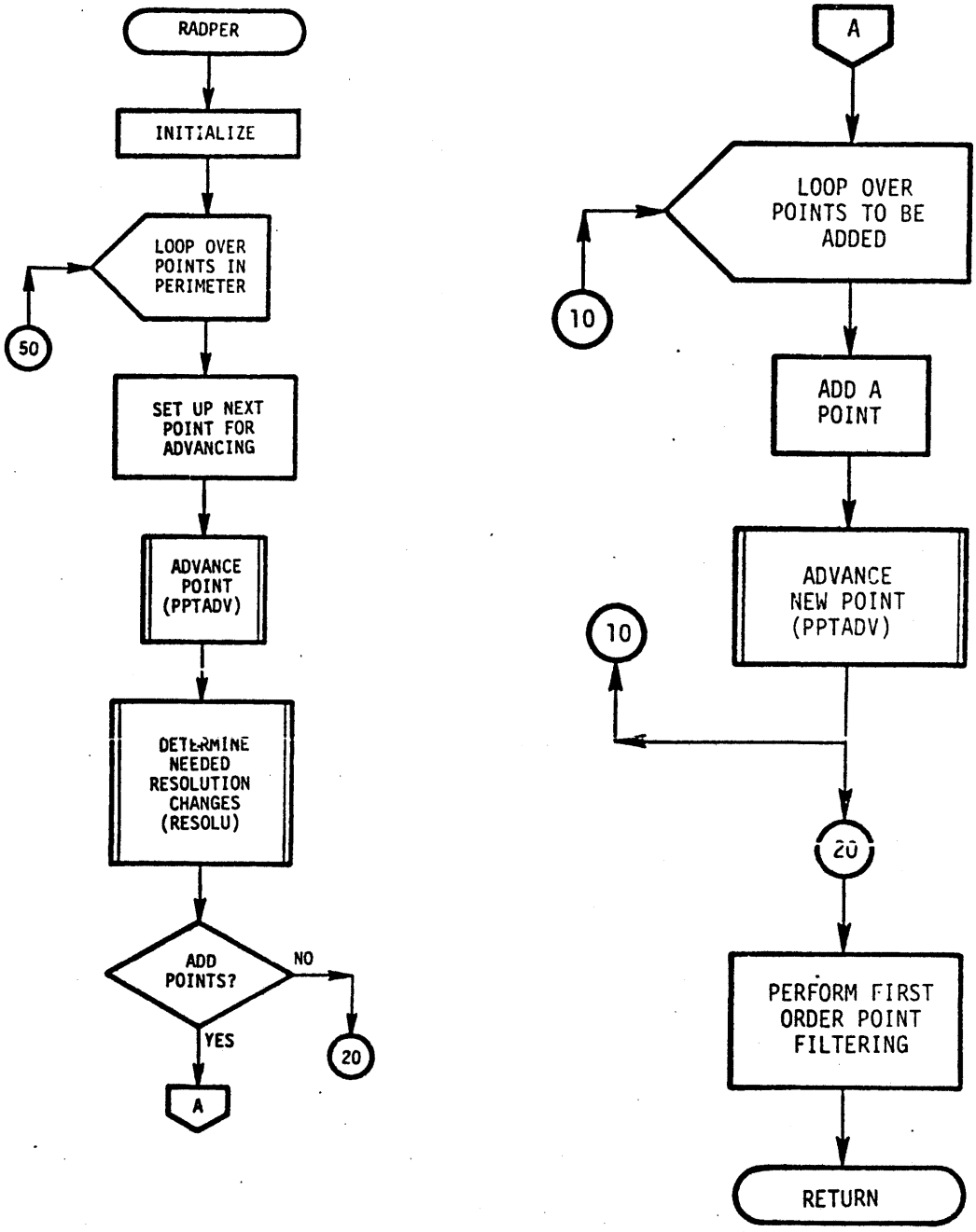


Figure 22

Radial Perimeter Growth

```
      SUBROUTINE RADPER(TOF,IDOF,PTLAST,TODMAX,SAMSIZ,XP,YP,IMAX,NPMX,
     + JMAX,J,DT,ROSMLP,ROSMNL,ROE0,ROEMN,NNPTS,DISP,POLANG,IAVAIL,
     + IACTIV,XINT,YINT,LINKF,LINKB,ROSMCP,ROSMNC)
C
C     THIS ROUTINE CALCULATES ONE NEW INTERNAL PERIMETER IN A LINKED
C     LIST FORM AND ELIMINATES FIRST ORDER CROSSINGS
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     TOF     - TIME OF FIRE IN MINUTES FROM MIDNIGHT.
C     IDOF    - DAY OF YEAR
C     PTLAST  - TIME OF LAST INTERNALLY CALCULATED PERIMETER
C     TODMAX  - MAXIMUM TIME OF DAY IN MINUTES (=1440)
C     SAMSIZ  - SAMPLE INTERMAL SIZE FOR PROPAGATING ALONG A LINE
C     XP      - X COORDINATES OF PERIMETERS. (SEE RADSM)
C     YP      - Y COORDINATES OF PERIMETERS. (SEE RADSM)
C     IMAX    - THE MAXIMUM NUMBER OF STORAGE LOCATIONS AVAILABLE FOR
C               ANY ONE PERIMETER
C     NPMX    - THE MAXIMUM NUMBER OF POINTS IN ANY ONE PERIMETER
C     JMAX    - THE MAXIMUM NUMBER OF PERIMETERS WHICH MAY BE STORED
C     J       - INDEX OF STARTING PERIMETER
C     DT      - TIME INCREMENT TO NEXT INTERNAL PERIMETER
C     ROSMLP  - MAXIMUM RATE OF SPREAD ON LAST PERIMETER
C     ROSMNL  - MINIMUM RATE OF SPREAD ON LAST PERIMETER
C     ROE0    - BASE RESOLUTION PARAMETER
C     ROEMN   - MINIMUM RESOLUTION PARAMETER
C         INPUTS/ OUTPUTS
C     NNPTS   - NUMBER OF POINTS IN PERIMETER J
C         OUTPUTS
C     DISP    - AN ARRAY WHICH CONTAINS THE DISTANCE A POINT MOVED ON
C               ITS LAST ADVANCE.  (LENGTH = IMAX)
C     POLANG  - AN ARRAY OF POLAR ANGLES WHICH INDICATE THE DIRECTION
C               A POINT MOVED ON ITS LAST ADVANCE.  (LENGTH = IMAX)
C     IAVAIL  - INDEX TO AN AVAILABLE STORAGE LOCATION LINKED LIST WHICH
C               WAS CREATED WHEN POINTS WERE REMOVED FROM THE LIST,
C               IF = 0 STORAGE AVAILABLE AT NEXT UNUSED LOCATION
```

```
C     IACTIV = INDEX TO THE ACTIVE INTERNAL PERIMETER POINTS LINKED LIS
C     XINT   = INTERNAL PERIMETER X COORDINATE LINKED LIST
C     YINT   = INTERNAL PERIMETER Y COORDINATE LINKED LIST
C     LINKF  = FORWARD LINK FOR XINT,YINT
C     LINKB  = BACKWARD LINK FOR XINT,YINT
C     ROSMCP = MAXIMUM RATE OF SPREAD ON CURRENT PERIMETER
C     ROSMNC = MINIMUM RATE OF SPREAD ON CURRENT PERIMETER
C
      COMMON/DEACT/KOMIT
      DIMENSION XP(IMAX,JMAX),YP(IMAX,JMAX),DISP(IMAX),POLANG(IMAX)
      DIMENSION LINKF(IMAX),LINKB(IMAX)
      DIMENSION XINT(IMAX),YINT(IMAX)
      REAL MI,ML
      LOGICAL PREMOV
C
C         STATEMENT FUNCTION  TO CALCULATE DISTANCE
      DIST(DX,DY)=SQRT(DX*DX+DY*DY)
C
      ICMAX=1
C
      ROSMCP=0.0
      ROSMNC=1.0E30
      HALFPI=3.141592/2.0
      TOPP=TOF+PTLAST
      IDOPP=IDOF+IFIX(TOPP/TODMAX)
      TOPP=AMOD(TOPP,TODMAX)
C
      XPREV=XP(NNPTS-1,J)
      YPREV=YP(NNPTS-1,J)
      XI=XP(NNPTS,J)
      YI=YP(NNPTS,J)
      IC=0
      XPP=XP(NNPTS-2,J)
      YPP=YP(NNPTS-2,J)
      ALPHA=THETAD(XPP-XPREV,YPP-YPREV)
      BETA=THETAD(XI-XP,YI-YP)
      CALL PPTADV(DT,SAMSIZ,TOPP,IDOPP,XP,YP,BETA,ALPHA,ANG,
     + XOUT,YOUT,ROSMI,ROSMNI)
      DO 50 I=1,NNPTS
      XN=XP(I,J)
      YN=YP(I,J)
C
      ALPHA=THETAD(XPREV-XI,YPREV-YI)
      BETA=THETAD(XN-XI,YN-YI)
      ANG=(ALPHA-BETA)+0.50000+BETA
      IF(ALPHA.LT.BETA)ANG=ANG+3.141592
      ICP=IC
      IC=IC+1
      LINKF(IC)=IC+1
      LINKB(IC)=ICP
C
      ROSMIL=ROSMI
      CALL PPTADV(DT,SAMSIZ,TOPP,IDOPP,XI,YI,BETA,ALPHA,ANG,
     + XINT(IC),YINT(IC),ROSMI,ROSMNI)
      POLANG(IC)=ANG
      IF(ROSMI.GT.ROSMCP)ICMX=IC
      ROSMCP=AMAX1(ROSMCP,ROSMI)
      ROSMNC=AMIN1(ROSMNC,ROSMNI)
      DISP(IC)=DIST(XI-XINT(IC),YI-YINT(IC))
      SI=DIST(XPREV-XI,YPREV-YI)
C  CAUTION ..... ROEI CAN NEVER BE ALLOWED TO BE ZERO
      CALL RESOLU(ROE0,ROEMN,NPMX,IC,I,NNPTS,SI,J,AMAX1(ROSMI,ROSMIL),
     + ROSMNL,ROSMLP,IPA)
      IF(IPA.EQ.0) GO TO 20
C         PRESERVE DATA FOR POINT I
```

```
      DM=DISP(IC)
      XM=XINT(IC)
      YM=YINT(IC)
      IF(IC.EQ.ICMX)ICMX=ICMX+IPA
      ROSMIL=ROSMI
C         BACK UP ONE POINT
      IC=IC-1
C         SETUP PARAMETERS FOR GENERATING EXTRA POINTS ON THIS SEGMENT
      DS=SI/FLOAT(IPA+1)
C
      BETA=THETAO(XI-XPREV,YI-YPREV)
      ALPHA=BETA+3.141593
      DX=DS*COS(BETA)
      DY=DS*SIN(BETA)
C
C     BEGIN LOOP TO GENERATE EXTRA POINTS ON A SEGMENT
C
      DO 10 L=1,IPA
      XL=XPREV+DX*FLOAT(L)
      YL=YPREV+DY*FLOAT(L)
      ICP=IC
      IC=IC+1
      CALL PPTADV(DT,SAMSIZ,TOPP,IDOPP,XL,YL,BETA,ALPHA,ANGOUT,
     + XINT(IC),YINT(IC),ROSMI,ROSMNI)
      POLANG(IC)=ANGOUT
      IF(ROSMI.GT.ROSMCP)ICMX=IC
      ROSMCP=AMAX1(ROSMCP,ROSMI)
      ROSMNC=AMIN1(ROSMNC,ROSMNI)
      DISP(IC)=DIST(XL-XINT(IC),YL-YINT(IC))
      LINKF(IC)=IC+1
      LINKB(IC)=ICP
   10 CONTINUE
C         STORE DATA FOR POINT I
      ICP=IC
      IC=IC+1
      LINKF(IC)=IC+1
      LINKB(IC)=ICP
      POLANG(IC)=PM
      DISP(IC)=DM
      XINT(IC)=XM
      YINT(IC)=YM
      ROSMI=ROSMIL
   20 XPREV=XI
      YPREV=YI
      XI=XN
      YI=YN
   50 CONTINUE
C         LINK ENDS TOGETHER
      LINKF(IC)=1
      LINKB(1)=IC
      NNPTS=IC
C         PRINT MAX SPREAD RATE AND POSITION
      WRITE(6,6500)ROSMCP,TOPP,XINT(ICMX),YINT(ICMX)
 6500 FORMAT(3X,20HMAXIMUM SPREAD RATE ,E15.7,6H TIME ,F6.0,
     + 5H POS ,2E15.7)
C
C     FIRST ORDER POINT FILTERING
C
      IAVAIL=0
      IACTIV=1
      IF(MOD(KOMIT,10).GT.0)RETURN
   52 PREMOV=.FALSE.
      LFI=IACTIV
      NPLOOP=NNPTS
      DO 58 LOOP=1,NPLOOP
      AI=POLANG(LFI)
```

```
      XI=XINT(LFI)
      YI=YINT(LFI)
      LFL=LINKB(LFI)
      AL=POLANG(LFL)
      XL=XINT(LFL)
      YL=YINT(LFL)
C         CALCULATE INTERSECTION, IF ANY
      SINAI=SIN(AI)
      SINAL=SIN(AL)
      COSAI=COS(AI)
      COSAL=COS(AL)
      IF(ABS(COSAI).LE.1.E-6)GO TO 60
      IF(ABS(COSAL).LE.1.E-6)GO TO 70
      MI=SINAI/COSAI
      ML=SINAL/COSAL
      DM=ML-MI
      IF(ABS(DM).LE.1.E-6) GO TO 57
      BI=YI-MI*XI
      X=(BI-(YL-ML*XL))/DM
      Y=MI*X+BI
      DAI=ABS(THETA0(XI-DISP(LFI)*COSAI-X,YI-DISP(LFI)*SINAI-Y)-
     + THETA0(XI-X,YI-Y))
      DAL=ABS(THETA0(XL-DISP(LFL)*COSAL-X,YL-DISP(LFL)*SINAL-Y)-
     + THETA0(XL-X,YL-Y))
      IF(DAI.LE.HALFPI.OR.DAL.LE.HALFPI)GO TO 57
C
C     REMOVE POINT WITH SHORTEST VECTOR
C
   54 JOUT=LFL
      IF(DISP(LFL).LE.DISP(LFI))GO TO 55
      JOUT=LFI
      LFI=LFL
C         REMOVE POINT JOUT
   55 IF(NNPTS-1.LT.3)GO TO 90
      NNPTS=NNPTS-1
      PREMOV=.TRUE.
      IPBL=LINKB(JOUT)
      IPAL=LINKF(JOUT)
      IACTIV=IPAL
      LINKF(IPBL)=IPAL
      LINKB(IPAL)=IPBL
      IF(IAVAIL.EQ.0)GO TO 56
      ILAST=LINKB(IAVAIL)
      LINKB(IAVAIL)=JOUT
      LINKF(JOUT)=0
      LINKF(ILAST)=JOUT
      LINKB(JOUT)=ILAST
      GO TO 57
   56 IAVAIL=JOUT
      LINKF(IAVAIL)=2
      LINKB(IAVAIL)=IAVAIL
      GO TO 57
C         LINE AT POINT I IS VERTICAL
   60 IF(ABS(COSAL).LE.1.E-6)GO TO 57
      ML=SINAL/COSAL
      X=XI
      BL=YL-ML*XL
      Y=ML*X+BL
      IF(SINAI.LT.0.0)GO TO 65
      IF(YI-DISP(LFI).GT.Y.OR.Y.GT.YI)GO TO 57
      GO TO 54
   65 IF(YI.GT.Y.OR.Y.GT.YI+DISP(LFI))GO TO 57
      GO TO 54
C         LINE AT POINT L IS VERTICAL
   70 MI=SINAI/COSAI
      X=XI
```

```
      BI=YI-MI*XI
      Y=MI*X+BI
      IF(SINAL.LT.0.0)GO TO 75
      IF(YL-DISP(LFL).GT.Y.OR.Y.GT.YL)GO TO 57
      GO TO 54
   75 IF(YL.GT.Y.OR.Y.GT.YL+DISP(LFL))GO TO 57
      GO TO 54
C         NO INTERSECTION
   57 LFI=LINKF(LFI)
   58 CONTINUE
      IF(PREMOV)GO TO 52
   80 RETURN
   90 WRITE(6,6000)NNPTS
 6000 FORMAT(6X,26H*ERROR* RAUPER LINE55 NNPTS=,I5)
      GO TO 80
      END
```

6.5 PERIMETER LOOP REMOVAL (MASSAG)

1. Purpose

This routine removes small erroneous loops from a calculated fire perimeter.

2. Arguments

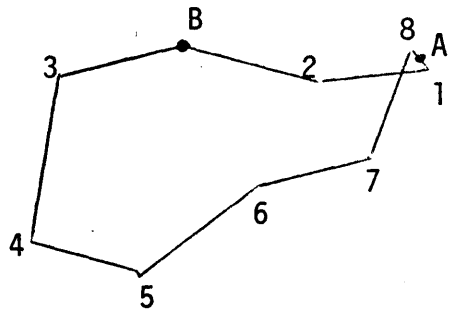
Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

This routine uses a floating grid to determine which line segments of the perimeter may intersect. Candidate segments are then tested for intersections. An intersection of two segments indicates a loop in the fire perimeter.[3]

The grid is used to determine the proximity of segments by recording all segments within each cell. The cells are recorded for each segment and each time a cell is found to contain another segment, the segments are checked to determine if they intersect. The entire process is performed using scatter storage techniques to minimize table size and table lookup time. In addition all tables are dynamically stored to further minimize storage requirements.

When an intersection is located, the loop to be deleted must be identified. The figure below indicates the potential problem. If the search around the perimeter for a loop starts at point B, the small loop containing point A will be found but if the search begins at point A, the large loop will be found. In general, loop positions are unknown; therefore, a test is made when a loop is detected to determine which loop is to be removed (RTLOOP).



Three operations occur when a loop is discovered. A test is made to determine which loop is to be removed, the selected loop is removed from the cell storage tables and the selected loop is removed from perimeter storage.

4. Comments

At the termination of this routine, the complete new perimeter is represented in a grided form as well as a linked list form. This could possibly be used in the future to determine perimeter area since the area inclosed in the perimeter equals the area of the grid less the area outside the perimeter but within the grid.

5. Flow Chart

See Figure 23.

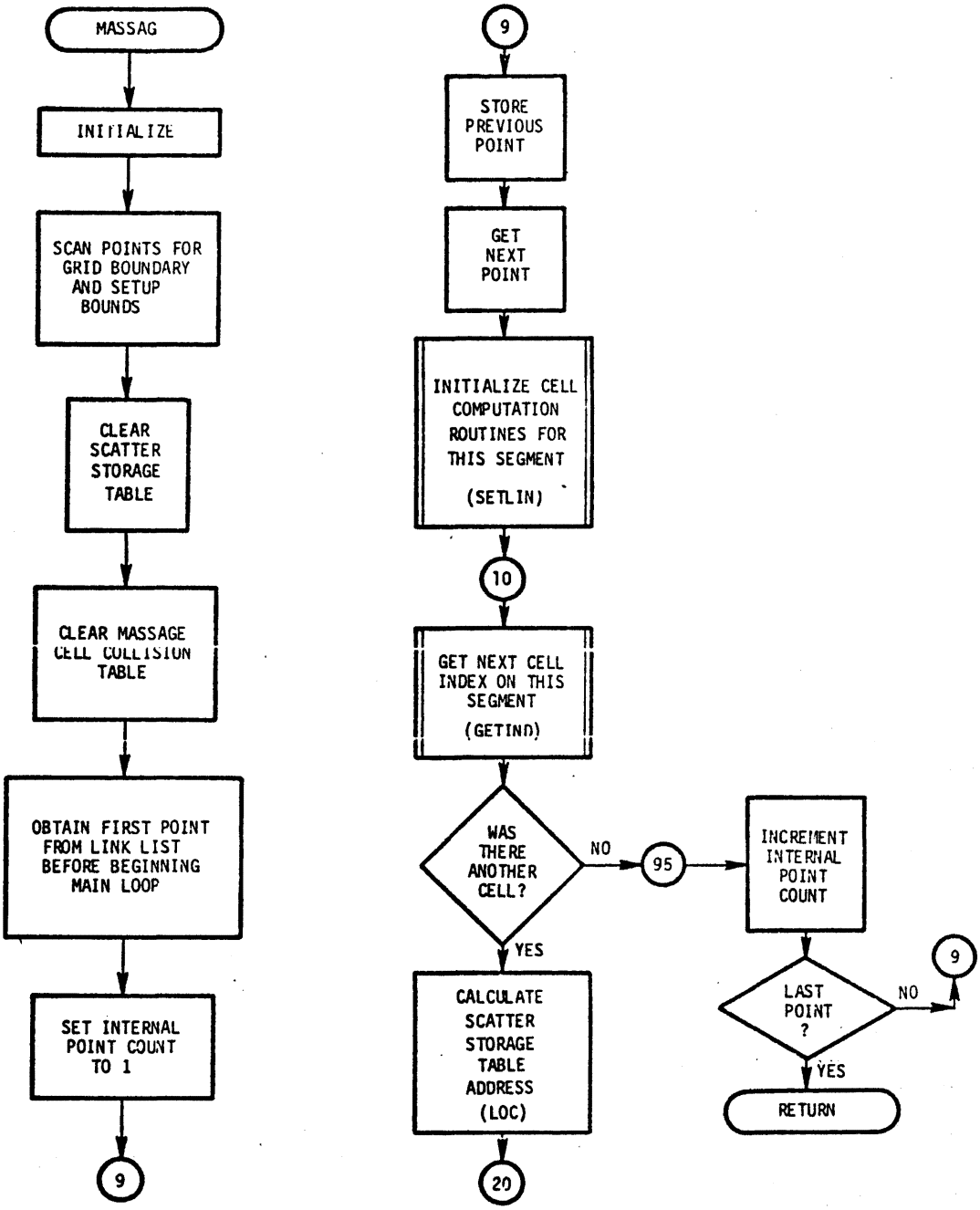


Figure 23

Loop Removal

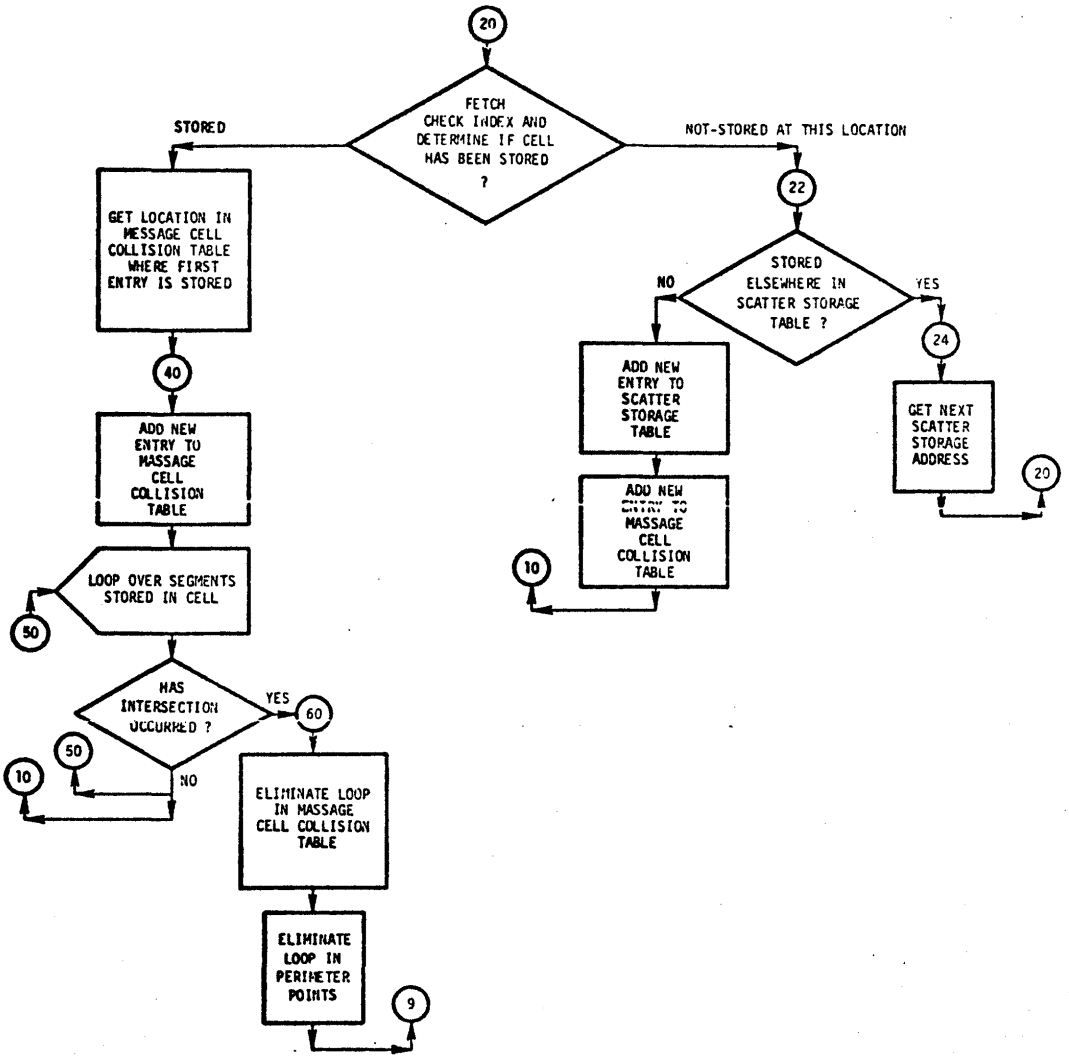


Figure 23

Loop Removal (Cont'd.)

```
      SUBROUTINE MASSAG(MXMASH,MXMCCT,ISS,IMAX,NORM,MARG,MASH,MCCT,XINT,
     + YINT,LINKF,LINKB,IACTIV,IAVAIL,NNPTS)
C
C     THIS ROUTINE MASSABES RADIAL PERIMETERS IN A LINKED - LIST FORM T
C     ELIMINATE LOOPS IN THE PERIMETER
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSUN, JR.
C
C         INPUTS
C     MXMASH - MAXIMUM LENGTH OF MASH TABLE
C     MXMCCT - MAXIMUM LENGTH OF MCCT TABLE
C     ISS    - SCATTER STORAGE FUNCTION
C     IMAX   - MAXIMUM NUMBER OF PERIMETER POINTS
C     NORM   - NORMAL MAXIMUM NUMBER OF CELLS
C     MARG   - NUMBER OF SCATTER STORAGE COLLISION CELLS
C         WORKING STORAGE FROM LIST
C     MASH   - SCATTER STORAGE TABLE
C              EACH SECTION CONTAINS
C                   1. CELL INDEX (VALUE = 1 TO 250,000 OR LARGER)
C                   2. INDEX TO MCCT
C                   3. NEXT CELL WITH THIS SCATTER STORAGE ADDRESS
C                      (=0 IF NONE)
C     MCCT   - MASSAGE CELL COLLISION TABLE
C              EACH SECTION CONTAINS
C                   1. INDEX OF SEGMENT CROSSING THIS CELL
C                   2. NEXT CROSSING SEGMENT, MCCT INDEX (=0 IF NONE)
C         INPUT/OUTPUTS
C     XINT   - X COORDINATES OF POINTS ON PERIMETER
C     YINT   - Y COORDINATES OF POINTS ON PERIMETER
C     LINKF  - FORWARD LINK
C     LINKB  - BACKWARD LINK
C     IACTIV - INDEX OF ONE POINT ON THE PERIMETER (ACTIVE STORAGE
C              POINTER)
C     IAVAIL - INDEX OF ONE POINT ON THE AVAILABLE STORAGE LIST
C              (=0 IF NONE)
C     NNPTS  - NUMBER OF POINTS ON PERIMETER
C
      COMMON /TABDAT/ NORMS,MSMCCT,MSMASH,XMING,XMAXG,YMING,YMAXG,
     + NX,NY,GRDSIZ,GRSIZI,NXNY,GRSQ2
      COMMON/DEACT/KOMIT
C
      DIMENSION MASH(MXMASH),MCCT(MXMCCT)
      DIMENSION XINT(IMAX),YINT(IMAX)
      DIMENSION LINKF(IMAX),LINKB(IMAX)
C
      LOGICAL NOTOK
      LOGICAL SWLOOP
      LOGICAL NOCROS
```

```
      EXTERNAL ISS
C
C          STATEMENT FUNCTION TO COMPUTE DISTANCE
C
      DIST(DX,DY)=SQRT(DX*DX+DY*DY)
C
C
C     INITIALIZATION
C
      NHOLD=NNPTS
      NXTMC=NORM*3+1
      NXTMA=0
      NEXTC=1
      NEXTA=0
C
C     SCAN ALL POINTS FOR MAXIMA AND MINIMA, CALCULATE AVERAGE
C     RESOLUTION (FOR REQUIRED GRID SPACING), AND ESTABLISH GRID
C
      IL=IACTIV
      NHOLD=MAX0(NHOLD,IL)
      X1=XINT(IL)
      Y1=YINT(IL)
      XMING=X1
      XMAXG=X1
      YMING=Y1
      YMAXG=Y1
      PDIST=0.0
      DO 2 I=1,NNPTS
      X2=X1
      Y2=Y1
      IL=LINKF(IL)
      NHOLD=MAX0(NHOLD,IL)
      X1=XINT(IL)
      Y1=YINT(IL)
      XMING=AMIN1(XMING,X1)
      XMAXG=AMAX1(XMAXG,X1)
      YMING=AMIN1(YMING,Y1)
      YMAXG=AMAX1(YMAXG,Y1)
      PDIST=PDIST+DIST(X2-X1,Y2-Y1)
    2 CONTINUE
      IL=LINKF(IL)
      PDIST=PDIST+DIST(X1-XINT(IL),Y1-YINT(IL))
      GRDSIZ=PDIST/FLOAT(NNPTS)
      GRSIZI=1.0/GRDSIZ
      GRSU2=GRDSIZ/2.0
      XMAXG=FLOAT(IFIX((XMAXG-XMING)*GRSIZI)+1)*GRDSIZ+XMING
      YMAXG=FLOAT(IFIX((YMAXG-YMING)*GRSIZI)+1)*GRDSIZ+YMING
      NX=IFIX((XMAXG-XMING+0.001)/GRDSIZ)
      NY=IFIX((YMAXG-YMING+0.001)/GRDSIZ)
      NXNY=NX*NY
C         CLEAR SCATTER STORAGE TABLE
      DO 4 I=1,MXMASH
      MASH(I)=0
    4 CONTINUE
C         CLEAR MASSAGE CELL COLLISION TABLE
      DO 6 I=1,MXMCCT
      MCCT(I)=0
    6 CONTINUE
C         STORE TABLE LIMITS FOR ISS
      NORMS=NORM
      MSMCCT=MXMCCT
      MSMASH=MXMASH
C         PREPARE TO BEGIN MAIN LOOP OVER POINTS
      IL=IACTIV
      XI=XINT(IL)
      YI=YINT(IL)
```

```
      IF(MOD(KOMIT/10,10).GT.0)RETURN
C
C
C     MAIN LOOP OVER POINTS IN PERIMETER
C
C
      IN = 1
    8 CONTINUE
C         SHIFT PREVIOUS POINT
    9 XPREV=XI
      YPREV=YI
      ILLAST=IL
C         GET NEXT POINT
      IL=LINKF(IL)
      XI=XINT(IL)
      YI=YINT(IL)
C         INITIALIZE CELL COMPUTATION ROUTINES
      CALL SETLIN (XPREV,YPREV,XI,YI)
C         GET NEXT CELL, IF ANY, ON LINE
   10 CALL GETIND(INDEX,NOTOK)
C         GO TO NEXT SEGMENT IF NO MORE CELLS ON THIS ONE
      IF(NOTOK)GO TO 95
C         CALCULATE SCATTER STORAGE TABLE ADDRESS
      LOC=(ISS(INDEX)-1)*3+1
C         FETCH CHECK INDEX FROM MASH
   20 ICHECK = MASH(LOC)
C         DETERMINE IF CELL HAS BEEN PREVIOULSY BEEN STORED.
      IF(ICHECK.NE.INDEX)GO TO 22
      LOCMT=MASH(LOC+1)
      GO TO 40
C     A SCATTER STORAGE COLLISTION MAY HAVE OCCURED
C     NOTHING MAY HAVE BEEN STORED
   22 ICOLL=MASH(LOC+2)
      IF(ICOLL.NE.0)GO TO 24
      IF(ICHECK.NE.0)GO TO 30
C     NO NEXT CELL EXIST IN MASH
C     NOTHING STORED AT LOC IN MASH
C     STORE INDEX IN MASH
   25 MASH(LOC)=INDEX
      IF(NEXTA.EQ.0)GO TO 27
C     AVAILABLE STORAGE EXISTS
      LOCMT=NEXTA
      NEXTA=MCCT(NEXTA+1)
C     CORRECT LOCATION FOUND IN MASH
C     STORE POINTER TO MCCT AND LOAD MCCT
   28 MASH(LOC+1)=LOCMT
      MCCT(LOCMT)=ILLAST
      MCCT(LOCMT+1)=0
      GO TO 10
   24 LOC = ICOLL
      GO TO 20
C     CORRECT LOCATION FOUND IN MASH ,  FIND LOCATION IN MCCT AND
C     UPDATE MCCT LOCATION COUNT
   27 IF(NEXTC.GT.MXMCCT)GO TO 200
      LOCMT=NEXTC
      NEXTC=NEXTC+2
      GO TO 28
C
C     CALCULATE LOCATION TO STORE COLLIDING CELLS
C
   30 CONTINUE
      WRITE(6,3006)INDEX,ICHECK
 3006 FORMAT(10X,17HCOLLISION, INDEX=,I5,2X,7HICHECK=,I5)
      IF(NXTMA.NE.0)GO TO 32
      IF(NXTMC.GT.MXMASH)GO TO 205
      MASH(LOC+2)=NXTMC
```

```
      LOC=NXTMC
      NXTMC=NXTMC+3
      GO TO 25
   32 LOC=NXTMA
      NXTMA=MASH(LOC+2)
      GO TO 25
C
C     STORE THIS SEGMENT (LINKB(IL)) UNDER THIS CELL (INDEX)
C
   40 IF(NEXTA.NE.0)GO TO 42
      IF(NEXTC.GT.MXMCCT)GO TO 210
      LOCFMT=LOCMT
      LOCMT=NEXTC
      NEXTC=NEXTC+2
   45 MASH(LOC+1)=LOCMT
      MCCT(LOCMT+1)=LOCFMT
      MCCT(LOCMT)=ILLAST
      GO TO 50
   42 LOCFMT=LOCMT
      LOCMT=NEXTA
      NEXTA=MCCT(LOCMT+1)
      GO TO 45
C
C     COMPARE THIS SEGMENT WITH ALL PREVIOUS SEGMENTS WHICH
C     ARE IN THIS CELL C
C
   50 ISEG=MCCT(LOCFMT)
      XPREV2=XINT(ISEG)
      YPREV2=YINT(ISEG)
      ISEGC=ISEG
      ISEG=LINKF(ISEG)
      XI2=XINT(ISEG)
      YI2=YINT(ISEG)
C          DETERMINE IF SEGMENT PREVIOUSLY STORED IS AJACENT
      IF(ISEG.EQ.ILLAST.OR.IL.EQ.ISEGC)GO TO 55
C  CAUTION ... DUPLICATE SEGMENTS NOT CONSIDERED INTERSECTIONS
      CALL INTERS(XPREV,YPREV,XI,YI,XPREV2,YPREV2,XI2,YI2,
     + XCROSS,YCROSS,NOCROS)
      IF(NOCROS)GO TO 55
      CALL SPIND(XCROSS,YCROSS,ICHECK)
      IF(ICHECK.EQ.INDEX)GO TO 60
   55 LOCFMT=MCCT(LOCFMT+1)
      IF(LOCFMT.NE.0)GO TO 50
      GO TO 10
C
C     ELIMINATE LOOP IN MCCT
C
C          INSURE REMOVAL OF CORRECT LOOP
   60 CALL RTLOOP(LINKF,IMAX,NNPTS,ISEG,ISEGC,IL,ILLAST,SWLOOP)
C          COUNT POINTS BEING REMOVED
      NPTSM=0
      ISEGI=ISEGC
      XI2=XINT(ISEGI)
      YI2=YINT(ISEGI)
C     GO TO NEXT SEGMENT IN LOOP BEING REMOVED
   64 ISEGP=ISEGI
      ISEGI=LINKF(ISEGI)
      XPREV2=XI2
      YPREV2=YI2
      YI2=YINT(ISEGI)
      XI2=XINT(ISEGI)
   62 NPTSM=NPTSM+1
      CALL SETLIN(XPREV2,YPREV2,XI2,YI2)
   63 CALL GETIND(INDEX2,NOTOK)
      IF(NOTOK)GO TO 64
```

```
      IF(ISEGP.EQ.ILLAST.AND.INDEX.EQ.INDEX2)NOTOK=.TRUE.
      LOC=(ISS(INDEX2)-1)*3+1
C         FIND CORRECT MASH ENTRY
      LOC1=LOC
   65 ICHECK=MASH(LOC)
      IF(ICHECK.EQ.INDEX2)GO TO 67
      LOCP=LOC
      LOC=MASH(LOC+2)
      IF(LOC.EQ.0)GO TO 80
      GO TO 65
   67 LOCMT=MASH(LOC+1)
C         IS THIS ENTRY TO BE ELIMINATED
      IF(MCCT(LOCMT).NE.ISEGP)GO TO 69
C         IS THIS THE ONLY ENTRY FOR THIS CELL
      IF(MCCT(LOCMT+1).NE.0)GO TO 70
C         ELIMINATE THE ONLY ENTRY (IN MCCT, AND CLEAR MASH)
      MCCT(LOCMT+1)=NEXTA
      NEXTA=LOCMT
      MASH(LOC)=0
      MASH(LOC+1)=0
C     WAS MASH LINKED THROUGH TO GET HERE
      IF(LOC.EQ.LOC1)GO TO 72
C     MASH WAS LINKED THROUGH
      MASH(LOCP+2)=MASH(LOC+2)
      MASH(LOC+2)=NXTMA
      NXTMA=LOC
      GO TO 80
C         ELIMINATE FIRST ENTRY (IN MCCT, ADJUSTING MASH)
   70 MASH(LOC+1)=MCCT(LOCMT+1)
      MCCT(LOCMT+1)=NEXTA
      NEXTA=LOCMT
      GO TO 80
C         FIRST CELL AT THIS SCATTER ADDRESS (IN MASH, TO BE ELIMINATED
C         BECAUSE ONLY ONE ENTRY EXISTED IN MCCT TABLE)
   72 IF(MASH(LOC+2).EQ.0)GO TO 80
   74 LOCN=MASH(LOC+2)
      MASH(LOC)=MASH(LOCN)
      MASH(LOC+1)=MASH(LOCN+1)
      MASH(LOC+2)=MASH(LOCN+2)
      MASH(LOCN)=0
      MASH(LOCN+1)=0
      MASH(LOCN+2)=NXTMA
      NXTMA=LOCN
      GO TO 80
C         REMOVE (FROM MCCT) THE ENTRY FOR THIS CELL HAVING BEEN CREATE
C         BY SEGMENT ISEGP
   69 LOCMTP=LOCMT
      LOCMT=MCCT(LOCMT+1)
C               THE NEXT TEST ELIMINATES PROBLEMS CAUSED BY FINDING AN
C               INTERSECTION BEFORE COMPLETING ALL CELL ENTRIES FOR A
C               SEGMENT AND ANY SEGMENTS NOT ENTERED IN A LOOP WHERE THE
C               FIRST POINT IS WITHIN THE LOOP
      IF(LOCMT.EQ.0.OR.LOCMTP.EQ.0)GO TO 80
      IF(MCCT(LOCMT).NE.ISEGP)GO TO 69
      MCCT(LOCMTP+1)=MCCT(LOCMT+1)
      MCCT(LOCMT+1)=NEXTA
      NEXTA=LOCMT
      GO TO 80
   80 IF(.NOT.NOTOK)GO TO 63
C         THE  CHANGES MADE TO MCCT AS A RESULT OF THE LOOP IN THE
C         PERIMERTER HAVE BEEN REMOVED AT THIS POINT
C
C     ELIMINATE LOOP IN PERIMETER POINTS
C
      IPAV=LINKB(ILLAST)
      XINT(ILLAST)=XCROSS
```

```
      YINT(ILLAST)=YCROSS
      LINKF(ISEGC)=ILLAST
      LINKB(ILLAST)=ISEGC
      IACTIV=ILLAST
C
C     LINK POINTS INTO AVAILABLE STORAGE
C
      IF(IAVAIL.EQ.0)GO TO 85
      ILAST=LINKB(IAVAIL)
      LINKB(IAVAIL)=IPAV
      LINKF(IPAV)=0
      LINKF(ILAST)=ISEG
      LINKB(ISEG)=ILAST
      GO TO 90
   85 IAVAIL=ISEG
      LINKF(IPAV)=0
      LINKB(IAVAIL)=IPAV
      GO TO 90
   90 NNPTST=NNPTST+1
      NNPTS=NNPTS-NPTSM+2
      IF(SWLOOP)GO TO 96
      IL=ISEGC
      XI=XINT(IL)
      YI=YINT(IL)
      GO TO 9
   95 IN=IN+1
      IF(IN.LE.NNPTST)GO TO 8
   96 CONTINUE
      WRITE(6,1000)NXTMC,NXTMA,NEXTC,NEXTA
      WRITE(6,1004)MXMASH,MXMCCT,NORM,MARG,IACTIV,IAVAIL,NNPTS
      IF(IN.GT.NNPTST-5000)RETURN
      WRITE(6,1005)(I,XINT(I),YINT(I),LINKF(I),LINKB(I),I=1,NHOLD)
      CALL DOUT(4HMASH,MASH,MXMASH,6)
      CALL DOUT(4HMCCT,MCCT,MXMCCT,8)
      RETURN
  200 WRITE(6,1001)NEXTC,MXMCCT
      GO TO 250
  205 WRITE(6,1002)NXTMC,MXMASH
      GO TO 250
  206 WRITE(6,1008)LOCMT,LOCMTP,ISEGP,MCCT(LOCMT),LOC,ICHECK,INDEX2
      GO TO 250
  210 WRITE(6,1003)NEXTC,MXMCCT
  250 WRITE(6,1004)MXMASH,MXMCCT,NORM,MARG,IACTIV,IAVAIL,NNPTS
      WRITE(6,1005)(I,XINT(I),YINT(I),LINKF(I),LINKB(I),I=1,NHOLD)
      CALL DOUT(4HMASH,MASH,MXMASH,6)
      CALL DOUT(4HMCCT,MCCT,MXMCCT,8)
      MXMASH=-IABS(MXMASH)
      RETURN
 1000 FORMAT(6X,16HMASSAGE COMPLETE/6X,6HNXTMC=,I5/6X,6HNXTMA=,I5/6X,
     + 6HNEXTC=,I5/6X,6HNEXTA=I5)
 1001 FORMAT(1H1,5X,7H*ERROR*//6X,21HNEXTC.GT.MXMCCT AT 27//
     + 6X,6HNEXTC=,I5,10X,7HMXMCCT=,I5/)
 1002 FORMAT(1H1,5X,7H*ERROR//6X,21HNXTMC.GT.MXMASH AT 30//
     + 6X,6HNXTMC=,I5,10X,7HMXMASH=,I5/)
 1003 FORMAT(1H1,5X,7H*ERROR*//6X,21HNEXTC.GT.MXMCCT AT 40//
     + 6X,6HNEXTC=,I5,10X,7HMXMCCT=,I5/)
 1004 FORMAT(1H0,5X,11HMASSAGE DMP//6X,7HMXMASH=,I5,10X,7HMXMCCT=,I5,
     + 10X,5HNORM=,I5,10X,5HMARG=,I5/6X,7HIACTIV=,I5,10X,7HIAVAIL=,I5,
     + 9X,6HNNPTS=,I5//8X,4HXINT,11X,4HYINT,11X,5HLINKF,3X,5HLINKB/)
 1005 FORMAT(1X,I5,2X,E12.5,3X,E12.5,3X,I5,3X,I5)
 1006 FORMAT(6X,4HMASH/1(4(5X,I5,1X,I9)))
 1007 FORMAT(6X,4HMCCT/1(4(5X,I5,1X,I9)))
 1008 FORMAT(1H1,5X,7H*ERROR*//6X,24HLOCMT OR LOCMTP =0 AT 69//
     + 6X,30HLOCMT,LOCMTP,ISEGP,MCCT(LOCMT)/6X,4I20/
     + 6X,17HLOC,ICHECK,INDEX2/6X,3I20)
      END
```

6.6 TIME INCREMENT CALCULATION (DLTRM)

1. Purpose

This routine calculates the time increment to the next perimeter.

2. Arguments

Each of the arguments is defined in the opening comments of the listir which follows.

3. Procedure

None. This is a dummy routine.

4. Comments

This routine is provided to allow the time increment to vary as a function of time or other variable(s) if required.

5. Flow Chart

Not required.

```
      FUNCTION DLTRM(TOF,IDOF,PTLR,PTLAST,DTMX)
C
C     THIS ROUTINE CALCULATES DELTA T FOR THE RADIAL SPREAD MODEL AS A
C     FUNCTION OF SEVERAL VARIABLES
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     TOF    - TIME OF FIRE IN MINUTES FROM MIDNIGHT
C     IDOF   - DAY OF FIRE IN DAYS FROM JAN 1
C     PTLR   - PERIMETER TIME IN MINUTES FROM TOF(IDOF) OF LAST REAL
C              PERIMETER
C     PTLAST - PERIMETER TIME IN MINUTES FROM TOF(IDOF) OF LAST
C              PERIMETER CALCULATED
C     DTMX   - DELTA T MAXIMUM VALUE
C         OUTPUT
C     DLTRM  - VALUE OF DELTA T TO BE USED
C
      DLTRM=DTMX
      RETURN
      END
```

6.7 INTERSECTION CALCULATION (INTERS)

1. Purpose

This routine determines if two line segments intersect, and the location of the point of intersection.

2. Arguments

INPUTS

X11	-	X coordinate of segment 1 point 1
Y11	-	Y coordinate of segment 1 point 1
X12	-	X coordinate of segment 1 point 2
Y12	-	Y coordinate of segment 1 point 2
X21	-	X coordinate of segment 2 point 1
Y21	-	Y coordinate of segment 2 point 1
X22	-	X coordinate of segment 2 point 2
Y22	-	Y coordinate of segment 2 point 2

OUTPUTS

XC	-	X coordinate of intersection
YC	-	Y coordinate of intersection
NØCRØS	-	Logical flag which indicates whether an intersection (crossing) has occurred (NOCROSS=TRUE means no crossing has occurred).

3. Procedure

The equations of each pair of line segments are calculated and solved simultaneously to define the point of intersection. A determination is then made to see if this point lies on both segments. Simplified equation forms are used for vertical and parallel line segments.

4. Comments

None.

5. Flow Chart

See Figure 24.

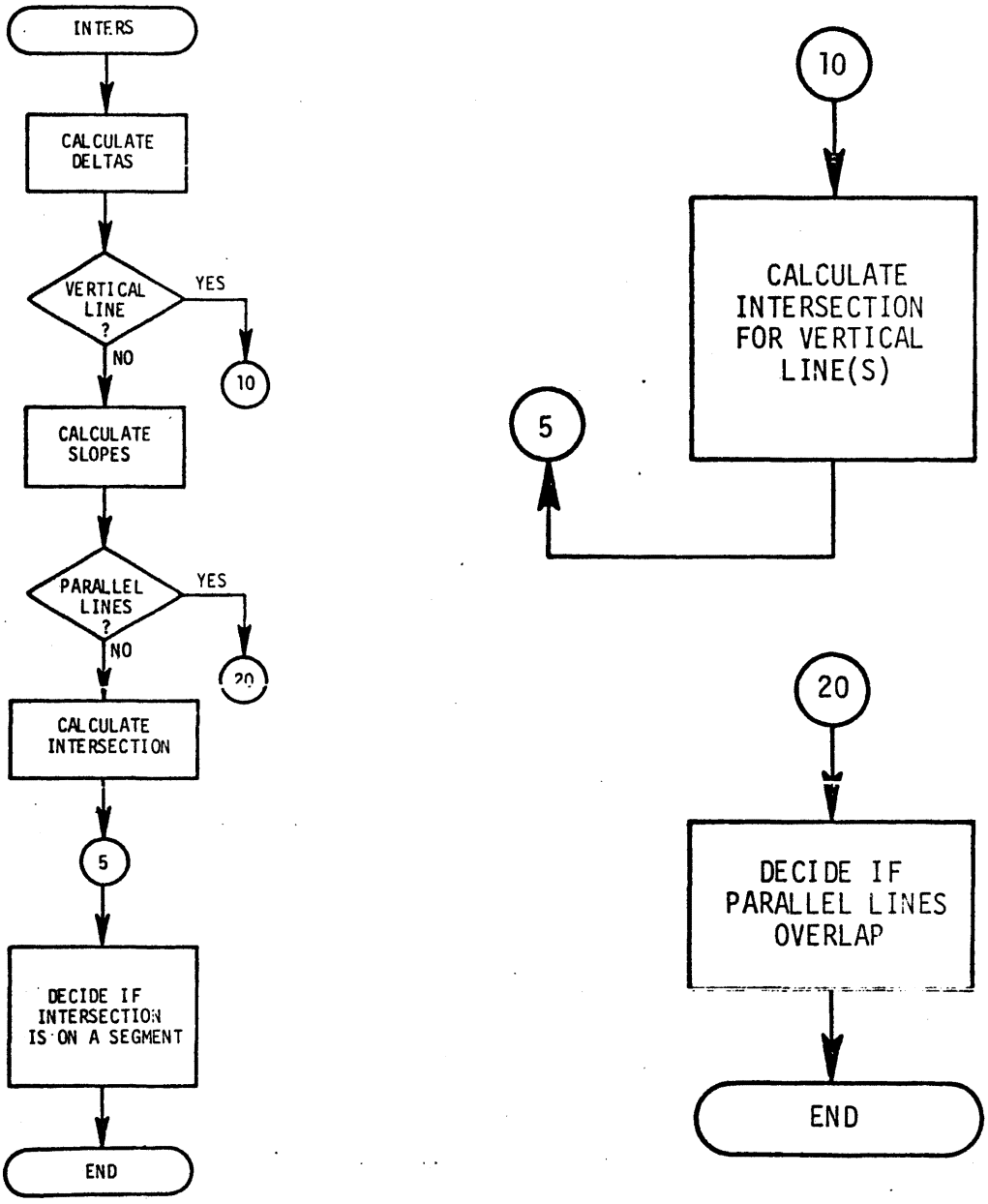


Figure 24

Intersection Calculation

```
      SUBROUTINE INTERS(X11,Y11,X12,Y12,X21,Y21,X22,Y22,XC,YC,NOCROS)
      LOGICAL NOCROS,SAME
C
      HALFPI=3.141592/2.0
      NOCROS=.FALSE.
      DX1=X12-X11
      DY1=Y12-Y11
      DX2=X22-X21
      DY2=Y22-Y21
      IF(ABS(DX1).LE.1.E-6)GO TO 10
      IF(ABS(DX2).LE.1.E-6)GO TO 15
      RM1=DY1/DX1
      RM2=DY2/DX2
      B1=Y11-RM1*X11
      B2=Y21-RM2*X21
      IF(RM1.EQ.RM2)GO TO 20
      XC=(B1-B2)/(RM2-RM1)
      YC=RM1*XC+B1
    5 DA1=ABS(THETAO(X11-XC,Y11-YC)-THETAO(X12-XC,Y12-YC))
      DA2=ABS(THETAO(X21-XC,Y21-YC)-THETAO(X22-XC,Y22-YC))
      IF(DA1.LE.HALFPI.OR.DA2.LE.HALFPI)NOCROS=.TRUE.
    6 RETURN
C         LINE 1 IS VERTICAL, IS LINE 2
   10 IF(ABS(DX2).LE.1.E-6)GO TO 35
C         ONLY LINE 1 IS VERTICAL
      RM2=DY2/DX2
      B2=Y21-RM2*X21
      YC=RM2*X11+B2
      XC=X11
      IF(YC.LT.AMIN1(Y11,Y12).OR.YC.GT.AMAX1(Y11,Y12))NOCROS=.TRUE.
      GO TO 5
C         ONLY LINE 2 IS VERTICAL
   15 RM1=DY1/DX1
      B1=Y11-RM1*X11
      YC=RM1*X21+B1
      XC=X21
      IF(YC.LT.AMIN1(Y21,Y22).OR.YC.GT.AMAX1(Y21,Y22))NOCROS=.TRUE.
      GO TO 5
C         BOTH SLOPES ARE SAME, ARE INTERCEPTS
      NOCROS=.TRUE.
      GO TO 6
C         BOTH SEGMENTS ARE PART OF SAME LINE, DO THEY OVERLAP
   25 X1MIN=AMIN1(X11,X12)
      X1MAX=AMAX1(X11,X12)
      XC=X21
      YC=Y21
      IF(X21.GE.X1MIN.AND.X21.LE.X1MAX)GO TO 5
      XC=X22
      YC=Y22
      IF(X22.GE.X1MIN.AND.X22.LE.X1MAX)GO TO 5
      NOCROS=.TRUE.
      GO TO 6
C         VERTICAL LINES, ARE THEY SAME
   35 IF(X11.EQ.X21)GO TO 45
      NOCROS=.TRUE.
      GO TO 6
C         BOTH SEGMENTS ARE PARTS OF SAME VERTICAL LINE, DO THEY OVERLA
   45 Y1MIN=AMIN1(Y11,Y12)
      Y1MAX=AMAX1(Y11,Y12)
      XC=X21
      YC=Y21
      IF(Y21.GE.Y1MIN.AND.Y21.LE.Y1MAX)GO TO 5
      XC=X22
      YC=Y22
      IF(Y22.GE.Y1MIN.AND.Y22.LE.Y1MAX)GO TO 5
      NOCROS=.TRUE.
      GO TO 6
      END
```

6.8 SPIRAL INDEX CALCULATION (SPIND AND ISPIND)

1. Purpose

This routine calculates the spiral index for a rectangular area, given the X,Y coordinates of a point within the area.

2. Arguments

INPUTS FROM CALLING LIST

X	-	X coordinate of input point
Y	-	Y coordinate of input point

INPUTS FROM COMMON/TABDAT/

NX	-	number of cells in the X direction
NY	-	number of cells in the Y direction
GRSIZI	-	length of a square cell side
XMING	-	minimum X grid value
YMING	-	minimum Y grid value
NXNY	-	the total number of cells in the grid

OUTPUT

ISPIRL	-	Spiral index of the cell containing the input point.

3. Procedure

The input point is converted to an X,Y index pair (IX,IY) by conventional means. This index pair is then converted to the spiral index by first calculating a level-number (LN), which is the number of complete spirals traversed plus one to arrive at the specified point. For example, in the sample grid shown in Figure 25, cell 39 is at level 3. The index of

NY = 7

17	16	15	14	13	12
18	33	32	31	30	11
19	34	41	40	29	10
20	35	42	39	28	9
21	36	37	38	27	8
22	23	24	25	26	7
1	2	3	4	5	6

NX = 6

Figure 25

Example of a Grid with Spiral Indices

the first cell on this level (LBI) is then calculated. In the example LBI is 37. The relative (spiral) position of the required cell may then be calculated relative to cell LBI.

4. Comments

SPIND and ISPIND are documented together and are logically one unit. The routines are coded separately since IX and IY are available by other means in SETLIN and GETIND.

5. Flow Chart

Not necessary.

```
      SUBROUTINE SPIND(X,Y,ISPIRL)
C
C     THIS ROUTINE CALCULATES THE SPIRIAL INDEX FOR A RECTANGULAR
C     AREA GIVEN A POINT
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
      COMMON /TABDAT/ NORMS,MSMCCT,MSMASH,XMING,XMAXG,YMING,YMAXG,
     + NX,NY,GRDSIZ,GRSIZI,NXNY,GRS02
C
      IX=IFIX((X-XMING)*GRSIZI)+1
      IY=IFIX((Y-YMING)*GRSIZI)+1
      CALL ISPIND(IX,IY,ISPIRL)
C
      RETURN
      END
```

```
      SUBROUTINE ISPIND(IX,IY,ISPIRL)
C
C     THIS ROUTINE CALCULATES THE SPIRIAL INDEX FOR A RECTANGULAR
C     AREA GIVEN A SET OF X-Y POINT INDICES
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
      COMMON /TABDAT/ NORMS,MSMCCT,MSMASH,XMING,XMAXG,YMING,YMAXG,
     + NX,NY,GRDSIZ,GRSIZI,NXNY,GRS02
C
      LN=MIN0(IABS(NX-IX)+1,IX,IABS(NY-IY)+1,IY)
      LNM1=LN-1
      LNX=NX-2*LNM1
      LNY=NY-2*LNM1
      LBI=NXNY-LNX*LNY+1
      LIX=IX-LNM1
      LIY=IY-LNM1
      IF(LIY.EQ.1)GO TO 10
      IF(LIY.EQ.LNY)GO TO 20
      IF(LIX.EQ.1)GO TO 30
      GO TO 40
   10 ISPIRL=LBI+LIX-1
      GO TO 50
   20 ISPIRL=LBI+2*LNX+LNY-LIX-2
      GO TO 50
   30 ISPIRL=LBI+2*LNX+2*LNY-LIY-3
      GO TO 50
   40 ISPIRL=LBI+LNX+LIY-2
C
   50 RETURN
      END
```

6.9 SCATTER STORAGE FUNCTION (ISS)

1. Purpose

This function maps a cell index into a storage location for scatter storage purposes.

2. Arguments

INPUTS FROM CALLING LIST

INDEX - The spiral cell index

INPUTS FROM COMMON/TABDAT/

NORMS - Normal maximum number of cells, a prime number (normally much greater than NPMX).

OUTPUT

ISS - Index to scatter storage

3. Procedure

A basic modulo function is employed.

4. Comments

None.

5. Flow Chart

Not Required.

```
      FUNCTION ISS(INDEX)
C
C     SCATTER STORAGE FUNCTION
C     VER 1.00
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
      COMMON /TABDAT/ NORMS,MSMCCT,MSMASH,XMING,XMAXG,YMING,YMAXG,
     + NX,NY,GRDSIZ,GRSIZI,NXNY,GRSQ2
C
      ISS=1+MOD(INDEX,NORMS)
C
      RETURN
      END
```

6.10 RESOLUTION ADJUSTMENTS (RESOLU)

1. Purpose

This routine determines segment resolution on the next calculated perimeter.

2. Arguments

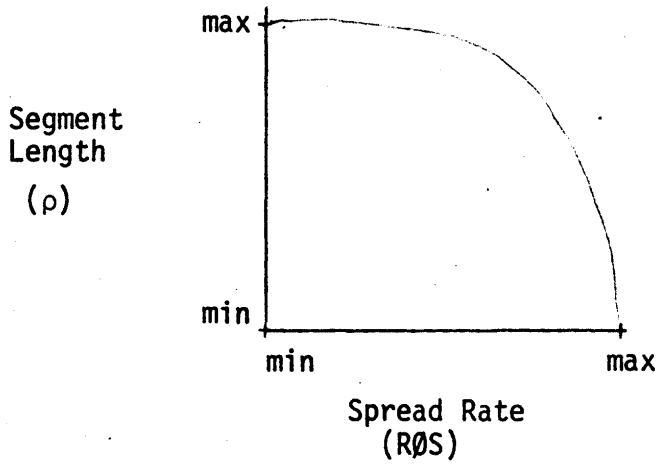
Each of the arguments is defined in the opening comments of the listing which follows, with the following exceptions:

INPUTS FROM COMMON/NEXTRS/

RA0	-	Change in spread rate between table entries
RA1	-	Desired spread rate range
RA2	-	Inverse of RA0
RFUNC	-	Resolution function table (length=21)
INTERL	-	Internal perimeter number

3. Procedure

The following approximate function is fit to the spread rates in the latest calculated perimeter.



This function is linearly interpolated to provide the required Segment Length (ρ). The number of points to be added is

$$IPA = IFIX(SI/\rho).$$

4. Comments

This routine returns the following values under the indicated conditions.

IPA Value	Condition
0	Point Saturation
1	First Internal Perimeter

5. Flow Chart

Not required.

```
      SUBROUTINE RESOLU(ROE0,ROEMN,NPMX,IC,I,NNPTS,SI,JJ,
     + ROSMI,ROSMNL,ROSMLP,IPA)
C
C     THIS ROUTINE DETERMINES THE RESOLUTION ALONG THE PERIMETER
C     VER 1.00
C
C    . PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     ROE0    - BASE RESOLUTION PARAMETER
C     ROEMN   - MINIMUM RESOLUTION PARAMETER
C     NPMX    - THE MAXIMUM NUMBER OF POINTS IN ANY ONE PERIMETER
C     IC      - THE CURRENT COUNT OF THE NUMBER OF POINTS GENERATED IN
C               THIS INTERNAL PERIMETER
C     I       - NUMBER OF POINTS CONSIDERED IN GENERATING THIS PERIMETER
C     NNPTS   - TOTAL POINTS TO BE CONSIDERED IN GENERATING THIS
C               PERIMETER
C     SI      - SIZE OF CURRENT SEGMENT
C     JJ      - INDEX OF CURRENT PERIMETER(INTERNAL)
C     ROSMI   - MAXIMUM SPREAD RATE AT CURRENT PERIMETER POINT I
C     ROSMNL  - MINIMUM RATE OF SPREAD ON PREVIOUS PERIMETER
C     ROSMLP  - MAXIMUM RATE OF SPREAD ON PREVIOUS PERIMETER
C         OUTPUT
C     IPA     - NUMBER OF POINTS TO BE ADDED TO THIS SEGMENT
      COMMON/DEACT/KOMIT
      COMMON /NEXTRS/ RA0,RA1,RA2,RFUNC(21),INTERL
C
      IPA=0
      IF(MOD(KOMIT/100,10).GT.0)GO TO 999
      ISIZE=NNPTS-I+IC
      IF(ISIZE.GE.NPMX)GO TO 999
      IPA=1
      IF(INTERL.LE.2)GO TO 999
      ROS=AMIN1(AMAX1(ROSMI,ROSMNL),ROSMLP)
      J=MIN0(IFIX((ROS-ROSMNL)/RA0)+2,21)
      X1=FLOAT(J-2)*RA0+ROSMNL
      Y1=ROEMN+RFUNC(J-1)*RA1
      YDIFF=(RFUNC(J)*RA1-Y1)
      ROE=Y1+YDIFF*(ROS-X1)*RA2
      ROE=AMAX1(ROE,ROEMN)
      IPA=IFIX(SI/ROE)
      IF(ISIZE+IPA.GE.NPMX)IPA=0
  999 RETURN
      END
```

6.11 SELECT RIGHT LOOP (RTLOOP)

1. Purpose

This routine selects the correct loop to be eliminated by MASSAG.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follws.

3. Procedure

The loop selection is made by counting the points in both potential loops. The smallest loop is chosen for elimination (see the documentation for MASSAG for an explanation of potential loops.)

4. Comments

None.

5. Flow Chart

Not required.

```
      SUBROUTINE RTLOOP(LINKF,IMAX,NNPTS,ISEG,ISEGC,IL,ILLAST,SWLOOP)
C
C     THIS ROUTINE INSURES THAT THE CORRECT LOOP IS ELIMINATED BY MASSA
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     LINKF  - FORWARD LINK ARRAY
C     IMAX   - SIZE OF FORWARD LINK ARRAY
C     NNPTS  - NUMBER OF POINTS CURRENTLY IN PERIMETER
C         INPUT/OUTPUTS
C     ISEGC  - INTERSECTED SEGMENT REAR POINT
C     IL     - CURRENT SEGMENT FORWARD POINT
C     ILLAST - CURRENT SEGMENT REAR POINT
C         OUTPUT
C     SWLOOP - LOGICAL VARIABLE TO INDICATE LOOP INDICIES SWAPPED (TRUE
C
      DIMENSION LINKF(IMAX)
C
      LOGICAL SWLOOP
C
C         SET COUNTER AND POSITION INDEX
      SWLOOP=.FALSE.
      NPTSM=-1
      ISEGI=ISEG
      ISEGP=ISEGC
C         COUNT POINTS IN LOOP
   10 NPTSM=NPTSM+1
C         HAS THE END OF THE LOOP BEEN REACHED
      IF(ISEGP.EQ.ILLAST)GO TO 20
C         STEP TO NEXT POINT
      ISEGP=ISEGI
      ISEGI=LINKF(ISEGI)
      GO TO 10
C
C     DETERMINE IF THIS IS A SMALL LOOP
C
   20 IF(NPTSM.LT.NNPTS/2)GO TO 30
C         WRONG LOOP - SWAP LOOPS
      ISEGI=ISEG
      ISEGP=ISEGC
      ISEG=IL
      ISEGC=ILLAST
      IL=ISEGI
      ILLAST=ISEGP
      SWLOOP=.TRUE.
C
   30 RETURN
      END
```

6.12 POSITIVE POLAR ANGLE (THETAO)

1. Purpose

This function calculates a positive polar angle.

2. Arguments

INPUTS

X	-	Change in X direction
Y	-	Change in Y direction

OUTPUTS

THETAO	-	A positive polar angle in the direction of X,Y

3. Procedure

The ATAN2 function is used to compute the angle, and the result is adjusted if negative. If X and Y are zero, the result is set to 10π.

4. Comments

None.

5. Flow Chart

Not required.

```
      FUNCTION THETA0(X,Y)
C
C     THIS FUNCTION CALCULATES A POSITIVE POLAR ANGLE
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
      THETA0=31.41593
      IF(X.NE.0.0.OR.Y.NE.0.0)THETA0=ATAN2(Y,X)
      IF(THETA0.LT.0.0)THETA0=THETA0+2.0*3.141592
C
      RETURN
      END
```

6.13 ERROR OUTPUT (DOUT)

1. Purpose

This routine prints the error output from MASSAG.

2. Arguments

INPUTS

ALPHA	-	Alpha description of output
MASH	-	Array to be output.
LENGTH	-	Length of MASH array
ISIZ	-	Words per line

OUTPUTS

A table is printed.

3. Procedure

The array is output at ISIZ words per line.

4. Comments

None.

5. Flow Chart

Not required.

```
      SUBROUTINE DOUT(ALPHA,MASH,LENGTH,ISIZ)
C
C     DEBUGGING OUTPUT COMPRESSION  ROUTINE
C
      DIMENSION MASH(LENGTH)
      WRITE(6,100)ALPHA
  100 FORMAT(2X,A10)
      ISIZM1=ISIZ-1
      L=((LENGTH-ISIZ)/ISIZ)*ISIZ
      IF(L.EQ.0)GO TO 11
      DO 10 I=1,L,ISIZ
      IP=I+ISIZM1
      DO 5 K=1,ISIZ
      IF(MASH(I+K-1).NE.0)GO TO 6
    5 CONTINUE
      GO TO 7
    6 WRITE(6,200)(J,MASH(J),J=I,IP)
    7 CONTINUE
  200 FORMAT(1(8(4X,I5,1X,I5)))
   10 CONTINUE
   11 L=L+1
      WRITE(6,200)(J,MASH(J),J=L,LENGTH)
      RETURN
      END
```

6.14 RESOLUTION SETUP (RESSET)

1. Purpose

This routine calculates three resolution parameters needed by RESOLU for the second and succeeding perimeters.

2. Arguments

INPUTS

RØE0	-	Base resolution parameter
RØEMN	-	Minimum resolution parameter
RØSMI	-	Maximum rate of spread on previous perimeter
RØSNI	-	Minimum rate of spread on previous perimeter
FACT	-	unused (=1.0)

OUTPUTS TO COMMON/NEXTRS/

RA0	-	Change in spread rate between table entries
RA1	-	Desired spread rate range
RA2	-	Inverse of RA0

3. Procedure

The quantities desired are calculated by the following formulas.

$$RA0 = (R\text{Ø}SMI - R\text{Ø}SNI)*0.05$$
$$RA1 = R\text{Ø}E0 - R\text{Ø}EMN$$
$$RA2 = 1.0/RA0$$

The number 0.05 is used since there are twenty table entries.

4. Comments

This routine is used in conjunction with RESOLU. Coding is included which was used at one time to produce line printer plots of the rescaled curv

5. Flow Chart

Not required.

```
      SUBROUTINE RESSET(ROE0,ROEMN,ROSMI,ROSNI,FACT)
C
C     THIS ROUTINE  CALCULATES THE THREE RESOLUTION PARAMETERS NEEDED B
C     RESOLU FOR THE SECOND AND SUCCEEDING PERIMETERS
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
      COMMON /NEXTRS/ RA0,RA1,RA2,RFUNC(21),INTERL
      DIMENSION X(23),Y(23)
C
      INTERL=INTERL+1
      IO=6
      IF(ROSMI.LE.ROSNI)WRITE(IO,100)ROSMI,ROSNI
  100 FORMAT(///6X,37H**ERROR**  ROSMLP.LE.ROSMNL IN RESSET/6X,7HROSMLP=
     * ,E20.5,10X,7HROSMNL=,E20.5///)
      RA0=(ROSMI-ROSNI)*0.05
      RA1=ROE0-ROEMN
      RA2=1.0/RA0
      WRITE(6,300)ROSMI,ROSNI
  300 FORMAT(2X,34HMAXIMUM AND MINIMUM SPREAD RATE = ,2E20.5)
      WRITE(IO,200)RA0,RA1,RA2
  200 FORMAT(6X,4HRA0=,E15.5,10X,4HRA1=,E15.5,10X,4HRA2=,E15.5)
      DO 80 I=1,23
      XI=RA0*FLOAT(I-2)+ROSNI
      XI=AMIN1(AMAX1(XI,ROSNI),ROSMI)
      X(I)=XI
      J=MIN0(IFIX((XI-ROSNI)/RA0)+2,21)
      X1=FLOAT(J-2)*RA0+ROSNI
      Y1=ROEMN+RFUNC(J-1)*RA1
      YDIFF=(RFUNC(J)-RFUNC(J-1))*RA1
      Y(I)=Y1+YDIFF*(XI-X1)*RA2
   80 CONTINUE
C     CALL LPPLOT(X,Y,23,1,1,.478,0.0,100.0,0.0,16HRESOLUTION CURVE,16,
C    * 11HSPREAD RATE,11,3HROE,3)
      RETURN
      END
```

REFERENCES

1. Rothermel, Richard C., "A Mathematical Model for Predicting Fire Growth in Wildland Fuels," USDA Forest Service Research Paper INT-115, 1972.

2. Sanderlin, James C., and Sunderson, John M., "Wildland Fire Modeling for Planning Support," Vol. II, "Initial Attack Assessment Models," Mission Research Corporation Report No. 7611-3-876, August 1976.

3. Sanderlin, James C. and Sunderson, John M., "A Simulation for Wildland Management Planning Support (FIREMAN)", Vol. IV, "Prototype Model Structure," Part I - "Major Structural Components," Mission Research Corporation Report No. 7612-6-1075, October, 1976 (pp. 204, 249).

4. FIRESCOPE Program Staff, "Preliminary Description of Fire Spread of the Potrero Fire," USDA Forest Service, Southwest Fire Research Laboratory, Riverside, California.